引用格式:李宛青,李树锋,刘健章,胡峰.基于深度Q网络的云演艺延迟敏感业务QoE优化[J].中国传媒大学学报(自然科学版),2024,31(01):49-55.
文章编号:1673-4793(2024)01-0049-07

基于深度Q网络的云演艺延迟敏感业务QoE优化

李宛青,李树锋,刘健章,胡峰*
(中国传媒大学信息与通信工程学院,北京100024)

摘要:网络中的资源分配问题一直备受关注,特别是在超高清视频流的传输中,对资源的有效管理至关重要。然而,随着网络服务的多样化和不断增加的业务类型,传统的资源分配策略往往显得不够灵活和智能。深度Q网络(Deep Q-Network,DQN)是一种能够自适应地学习和调整资源分配策略的神经网络模型。它基于神经网络与*Q*-Learning算法,通过不断尝试和学习来决策最佳的资源分配方案。本文旨在研究一种在云演艺场景下基于深度Q网络的延迟敏感业务资源调度算法,以满足当今网络中多样化的业务需求。仿真结果表明,基于深度Q网络的延迟敏感业务资源调度算法使得用户体验质量(Quality of Experience)指标显著提升,表明所提算法能够更好地满足延迟敏感业务的需求。

关键词:深度Q网络;资源调度;延迟敏感业务;用户体验;网络资源分配
中图分类号:TN919 **文献标识码:**A

QoE optimization of delay-sensitive cloud performance services based on Deep Q-Network

LI Wanqing, LI Shufeng, LIU Jianzhang, HU Feng*
(School of Information and Communication Engineering, Communication University of China, Beijing 100024, China)

Abstract: The problem of resource allocation in the network has been paid much attention, especially in the transmission of ultra-high-definition video streams, so the effective management of resources is very important. However, with the diversification of network services and the increasing types of business, the traditional resource allocation strategy often appears to be not flexible and intelligent enough. Deep Q-Network (DQN) is a kind of neural network model which can learn and adjust resource allocation strategy adaptively. It is based on the neural network and *Q*-Learning algorithm, through continuous trial and learning to decide the best resource allocation scheme. This paper aimed to study a delay-sensitive service resource scheduling algorithm based on DQN in the cloud performing arts scene, so as to meet the diversified service requirements in today's networks. Simulation results show that the delay-sensitive service resource scheduling algorithm based on DQN can significantly improve the Quality of Experience (QoE), indicating that the proposed algorithm can better meet the needs of delay-sensitive services.

Keywords: DQN; resource scheduling; delay-sensitive service; QoE; network resource allocation

基金项目:国家重点研发计划项目(2021YFF0900702)
作者简介(*为通讯作者):李宛青(1999-),女,硕士研究生,主要从事5G/6G与智能媒体通信研究。Email:liwanqing@cuc.edu.cn;胡峰(1985-),男,高级工程师,主要研究5G/6G通信、视觉物联网。Email:fenghu@cuc.edu.cn

1 引言

随着互联网的普及,多种多样的业务应用不断涌现,超高清视频流、云计算、物联网等领域的快速增长给网络资源的高效分配提出了新的挑战[1]。随着信息技术的不断发展和普及,云计算作为一种高效、灵活的计算模式逐渐渗透到了各行各业。其中,云演艺作为云计算的一个重要应用领域之一,为用户提供了多样化、便捷的娱乐体验[2]。然而,云演艺服务中延迟敏感业务的体验质量(Quality of Experience,QoE)受到了延迟问题的严重影响。演艺延迟敏感业务是指需要低延迟、高实时性的音视频传输和交互式应用,如在线直播、视频会议、互动游戏等,高延迟会导致视频卡顿、声音不同步等问题,从而影响用户观看体验[3]。当前,尽管云计算技术已经取得了长足的发展,但在云演艺延迟敏感业务的QoE优化方面仍存在着挑战。传统的网络优化方法往往难以有效地适应云演艺业务的特点,特别是在延迟敏感和大规模用户同时访问的情况下。因此,寻找一种更加有效、智能的优化方法来改善云演艺延迟敏感业务的QoE,是当前云计算领域急需解决的问题之一[4]。本文重点关注以块(block)形式发送数据的延迟敏感应用程序。

神经网络(Neural Network,NN)由大量人工神经元组成,能够接受来自外部环境或数据源的输入信息,通过前向传播和反向传播过程进行训练。Q学习算法(*Q*-Learning)迭代更新*Q*值函数,通过估计在每个状态下采取每个动作的长期回报值,最终获得最优策略。在多业务资源调度研究领域,将神经网络与Q学习算法相结合受到了广泛关注。Chmieliauskas等[5]提出了一种基于深度强化学习与*Q*-Learning的蜂窝网资源分配算法,以最大限度实现有限频谱资源的合理利用。他们的研究旨在通过智能资源分配来降低延迟和提高用户体验。实验结果显示,他们的算法与其他算法相比收敛速度有所提高,并在传输速率与优化能耗方面有明显改进,有效解决在多目标条件下的资源调度问题。因此,本文将神经网络作为Q学习的函数逼近器,实现了在复杂环境中学习并优化多业务资源块分配策略。

深度Q网络(Deep Q-Network,DQN)是深度强化学习(Deep Reinforcement Learning,DRL)中的一种重要方法,已在众多领域取得了显著成果,包括图像处理、自动驾驶、游戏和自然语言处理。Burhanuddin等[6]在无人机到无人机场景下探索了一种基于Actor-Critic(AC)与DQN的DRL算法,并通过仿真验证此算法在QoE方面优于Greedy算法,最终实现了稳定的视频传输与长期优化。

深度Q网络结合了神经网络和*Q*-Learning算法,具有自适应性和智能性,能够根据不同业务的需求进行资源分配决策[7]。此方法在资源分配问题中具有广泛的应用潜力,特别是在延迟敏感业务领域[8-10]。

针对以上问题,本文旨在基于DQN等深度强化学习技术,探索并提出一种针对云演艺延迟敏感业务资源调度算法,使得模型能够自适应地根据不同业务需求进行资源分配,提高资源利用率和用户体验。另外,本文提出了一种在深度强化学习中将神经网络用于近似*Q*值函数的方法,利用神经网络架构和资源估计技术,从实时网络信息中学习并动态调整资源的调度。这一算法在资源分配决策上具有灵活性和智能性,能够为不同类型的业务提供优化的服务。仿真结果表明,与Reno算法、Actor-Critic算法相比,本文所提算法可以有效提高用户体验质量。

2 资源调度环境架构

2.1 典型环境架构

图1展示了多业务资源调度场景下的典型环境架构,其中有数个发送方和接收方共享一个链接。根据不同应用程序数据集生成数据块,经过块调度模块与带宽估计模块进行调度后,以数据包的形式离开发送方进入链路。在这一过程中,链路的总可用带宽、最小延迟均根据网络跟踪做出相应变化。如果数据到达时刻的速率超过带宽,等待被发送的数据包将被临时储存在队尾的先入先出(First Input First Output,FIFO)队列中。当最后一个数据包到达接收方时,就可以计算这一业务块的完成时间,最终将其在模拟器内用于QoE计算。

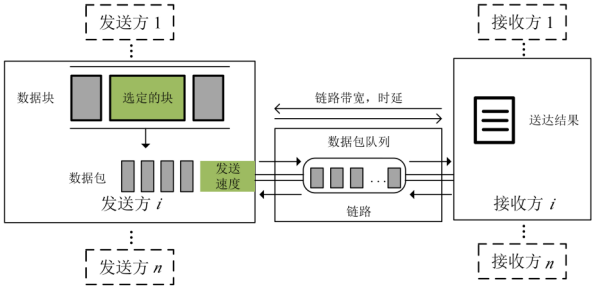

图1 多业务资源调度场景典型环境架构

为了保证实验环境支持数据块形式传输,并为数据块增添有效传输时限,本文使用ACM 2021 Multimedia Grand Challenge开源模拟器,数据块递送模拟器架构如图2所示,其中主要包括三部分:调度方案、环境、QoE模型。结合图1,在模拟之前根据环境设置

多个发送方与接收方，二者经过链路进行数据的传递。每个发送方在每条链路中模拟一个应用程序跟踪列表，每条链路中基于不同的网络跟踪进行相应的模拟，其中应用程序跟踪用于模拟应用程序的数据模式，网络跟踪用于模拟当前网络条件。

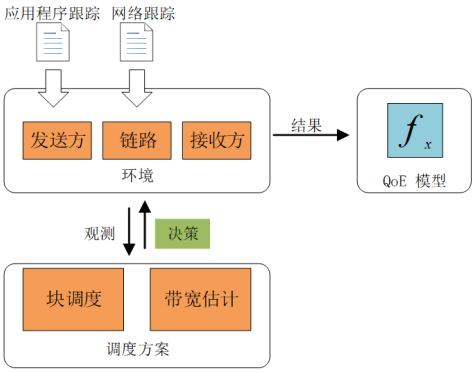

图2 数据块递送模拟器架构

2.2 QoE模型

在网络通信或数据传输领域，QoE通常用于评估用户对于服务、应用或内容的整体体验和满意度。多媒体应用QoE通常会由于所选方法的不同发生显著的变化，当数据块及时到达时，用户感知的质量通常得到改善。本文在基于数据块的优先级对QoE进行计算的基础上，考虑了数据块的紧急程度，将QoE建模为式(1)：

$$\mathrm{QoE}=\sum_{i=1}^{n}(x\times priorities[i]+(1-x)\times urgency[i]) \tag{1}$$

其中x为用于平衡优先级和紧急程度的权重参数，默认值为0.9。*priorities*是每个数据块的优先级和权重列表，*urgency*表示数据块的紧急程度。通过调整x的值，可以加强或减弱优先级和紧急程度对QoE的影响。

3 资源调度算法

3.1 拥塞控制算法

3.1.1 Reno算法

拥塞控制算法Reno是一种用于传输控制协议(Transmission Control Protocol，TCP)的算法，用于控制数据包在网络中传输的速率。Reno算法基于拥塞避免和快速恢复的概念，通过动态调整发送数据包的数量来维持网络的稳定性[11]。

该算法主要包括三种状态：慢启动(Slow Start)、拥塞避免(Congestion Avoidance)和快速恢复(Fast Recovery)。在慢启动阶段，拥塞窗口(*cwnd*)设为初始值，并以指数增长的方式逐渐增加，直到达到慢启动阈值(*ssthresh*)，*ssthresh*与*cwnd*计算如式(2)：

$$\begin{cases} ssthresh=cwnd/2, cwnd=1 & \text{超时} \\ ssthresh=cwnd/2, cwnd=ssthresh & \text{收到三个重复 } ACK \end{cases} \tag{2}$$

其中*ACK*为标志位，用来确认序号有效。对*ssthresh*进行减半处理，*ssthresh*与*cwnd*的表达式如式(3)、式(4)：

$$ssthresh=(1-\alpha)\times cwnd \tag{3}$$

$$cwnd=(1-\alpha)\times cwnd \tag{4}$$

其中α作为调节参数，取0.5。如果发生丢包，算法会进入快速恢复状态，然后根据情况调整阈值和*cwnd*，最终恢复到拥塞避免状态。算法过程描述如表1：

表1 Reno拥塞控制算法过程描述

Algorithm1：Reno拥塞控制算法
Step1：初始化拥塞窗口*cwnd*大小、慢启动阈值*ssthresh*等参数与状态变量；
Step2：以指数增加拥塞窗口*cwnd*的大小；
Step3：当*cwnd*达到*ssthresh*，算法进入拥塞避免状态，*cwnd*以线性方式增长，网络发生拥塞时触发快速恢复状态；
Step4：将*ssthresh*设置为当前*cwnd*值的一半，将*cwnd*设置为*ssthresh*，每收到一个确认，增加*cwnd*的大小一个最大报文段长度(MSS)，重复Step4直到回到Step3；
Step5：更新*cwnd*大小，动态调整算法状态和参数。

3.1.2 Actor-Critic算法

Actor-Critic的架构包括两个部分：

·策略网络Actor：观察网络状态，结合数据传输速率、丢包情况等输出选择动作，包括增加、保持、减少发送速率；

·评论网络Critic：评估Actor网络选择动作的价值，指导Actor网络更新策略，以获得更好的决策[12]。

伪代码如表2：

表2 Actor-Critic算法伪代码

Algorithm2：Actor-Critic算法
初始化神经网络参数θ_{actor}，θ_{critic}
循环EPISODES:
初始化环境状态s
初始化奖励总和t_ω
循环MAX_TIMESTEPS中时间步：
根据环境状态选择动作
获得奖励ω和新状态s'
估计动作价值函数$V(s)$：
$\delta_t=\omega+\gamma V(s')-V(s)$
更新Critic神经网络参数θ_{critic}
计算动作概率
计算Actor损失函数
更新Actor神经网络参数θ_{actor}
更新$s\leftarrow s'$
累加奖励$t_\omega+=\omega$
终止训练

3.2 资源调度算法

Reno算法作为TCP协议中的一种经典算法,基于拥塞窗口大小的动态调整来控制网络的拥塞状态。其主要缺陷之一是在高延迟网络中表现不佳,并且在面对长距离或高带宽延迟积(Bandwidth-Delay Product,BDP)的网络时容易出现性能下降。

这种性能下降部分源于Reno算法在拥塞控制中对丢包的处理。当出现丢包时,Reno算法采用指数退避的策略,即将拥塞窗口减半,并采用线性增长的方式重新开始。然而,对于高延迟网络来说,丢包往往被视为网络拥塞的信号,这导致了Reno算法过于保守的行为,因此未能充分利用网络带宽。

为了克服这些缺陷,本文研究了一种基于强化学习的动态拥塞控制方法。该方法利用神经网络和Q学习来自适应地调整发送速率,不仅能更好地适应网络状况的变化,还能在一定程度上减少对丢包的过度敏感。算法过程描述如表3:

表3 资源调度算法过程描述

Algorithm3:资源调度算法
Step1:神经网络估计发送速率,DQN输入状态,输出Q值;
Step2:存储并随机抽样历史经验,存储先前状态、动作与奖励;
Step3:基于Q值选择最优动作,更新神经网络模型并学习估算Q值;
Step4:估计当前可用带宽,从而调整发送速率。

DQN是基于深度学习的Q-learning算法,将深度神经网络技术与价值函数相结合,采用经历回放、目标网络的方法进行网络训练。Q-function函数可用式(5)描述:

$$Q_\pi(s_t,a_t)=E_\pi[r_t+\gamma Q_\pi(s_{t+1},a_{t+1})] \tag{5}$$

其中s_t为当前时间片的状态,a_t为采取的行为,r_t为奖励,s_{t+1}为新的状态,E_π表示以策略π进行动作。Q-function函数的最优值函数为式(6):

$$Q^*_\pi(s_t,a_t)=E_\pi[r_t+\gamma \max Q^*_\pi(s_{t+1},a_{t+1})] \tag{6}$$

对$\bar{Q}(s_t,a_t)$进行增量式更新时,使用时序差分[13]学习目标,如式(7):

$$\bar{Q}(s_t,a_t)=\bar{Q}(s_t,a_t)+\alpha[r_t+\gamma \max \bar{Q}(s_{t+1},a_{t+1})-\bar{Q}(s_t,a_t)] \tag{7}$$

其中$\bar{Q}$为函数逼近器,$\alpha\in(0,1]$为学习率,γ为折扣因子(衰减系数)。

得到$\bar{Q}$后可以获得Q^*_π的高质量表示,在时间步长t和状态s_t的条件下,从最优策略中对最优动作a_t进行采样,即$a_t\sim\pi^*(s_t)$,可用式(8)表示:

$$\pi^*(s_t)=\max \bar{Q} \tag{8}$$

对于连续状态与动作空间,可以使用NN来逼近最优动作值函数Q^*_π,在参数为θ时,$\bar{Q}=Q_\theta$。这时,可以将式(7)所表示的迭代过程视为回归问题,目标是通过上升随机梯度,估计NN的参数θ。在深度Q网络中,使用值$r_t+\gamma \max Q_{\theta target}$更新$Q_\theta$,其中$Q_{\theta target}$为目标$Q$-function。对于数据$[s_t,a_t,r_t,s_{t+1}]$,$Q$-function的损失函数构造为均方误差的形式,可由式(9)表示:

$$\arg\min_\theta \frac{1}{2N}\sum_{i=1}^{N}[Q_\theta(s_{t,i},a_{t,i})-(r_{t,i})+\gamma \max_{a_{t+1}} Q_\theta(s_{t+1,i},a_{t+1,i})]^2 \tag{9}$$

神经网络与Q-learning相互结合,神经网络用于逼近Q值函数。智能体通过与环境互动,收集数据并将其用于更新神经网络的权重,从而改善Q值函数的估计。这个过程将神经网络中的权重调整为最大化长期奖励,从而使算法能够做出更好的决策。

这种基于强化学习的方法通过持续学习和调整来实现自适应的拥塞控制,不仅在网络变化频繁的情况下表现更好,而且对丢包的反应更加灵活。相比于Reno算法,这种方法更能适应各种网络环境,提高了网络的传输效率和性能。

3.3 块选择算法

块选择算法通过权衡不同的数据块属性来决定下一个发送的数据块。在本文中,每个数据块都包含优先级、截止时间等基本属性。算法首先根据创建时间确定发送顺序,即先创建的数据块先被发送;如果多个数据块的创建时间相同,则选择剩余时间周期与截止交付时间比例更高的数据块作为优先发送对象。若数据块b_i的创建时间为T_{create_i},截止交付时间为$T_{deadline_i}$,当前时间节点为T_{cur},剩余时间周期与截止交付时间比例为R_i,则如式(10):

$$R_i=\frac{T_{deadline_i}-T_{cur}}{T_{deadline_i}-T_{create_i}},\text{如果}\ T_{cur}-T_{create_i}<T_{deadline_i} \tag{10}$$

块选择算法的设计能够有效管理本文系统中大量待发送的数据块,根据时间属性、剩余时间周期与截止交付时间的比例,对多种属性进行权衡,决定某一时刻应发送的数据块,从而提高了数据传输的效率和系统的整体性能。

4 仿真结果分析

仿真中涉及的参数值如表4所示。

为了验证本文提出的资源调度算法的有效性,本文数据集由三部分组成:块轨迹、背景流量轨迹和网络轨迹。每个应用场景都包含一个或多个块轨迹,这

表4 仿真参数

仿真参数	数值
动作频率	50
特征数量	1+2+2+50
动作数量	3
初始随机选择动作概率	0.9
学习率	0.01
折扣因子	0.9
目标网络参数更新频率	100
经验回放记忆容量	500
批量学习样本数	32
总步数	20000

些块轨迹可以与任意网络轨迹组合，模拟在特定网络条件下使用延迟敏感应用程序的过程。

图3以云演艺场景中实时通信（Real-Time Communications，RTC）应用程序为例，将来自RTC应用程序的流数据分为三类：第一类是控制信号，如预测带宽、目标比特率设置等，控制信号必须及时到达，才能保证RTC应用服务的稳定运行；第二类是音频，即剔除噪声后的用户语音数据；第三类是摄像机录制的视频。这三种类型的数据具有不同的优先级。错过控制信号的交付时间可能会导致QoE的严重下降，因此这些信号具有最高的优先级。在大多数RTC应用程序中，音频比视频更重要。

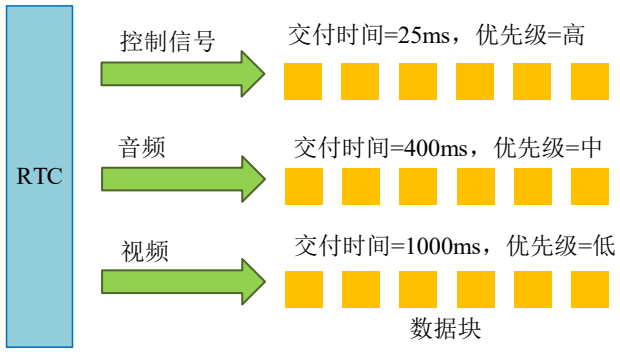

图3 RTC应用框架

图4、图5与图6分别展示了Reno算法、Actor-Critic算法作为拥塞控制模块与本文所提算法数据包的传输时延、平均传输时延。本文算法平均传输时延（L_{avg}）为0.044990s，将Reno算法、Actor-Critic算法作为拥塞控制模块的资源调度算法时，获得的平均传输时延分别为0.048450s与0.0484375s；本文算法用户体验质量（Q_{qoe}）达到了270，将Reno算法作为拥塞控制模块时的资源调度算法则为212。在相同的时间段内，由于本文算法适当减少对丢包的过度敏感，因此获得了更低延迟。

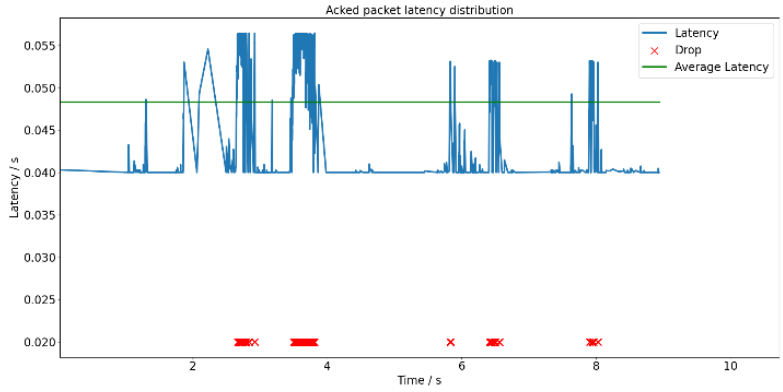

图4 Reno算法时延

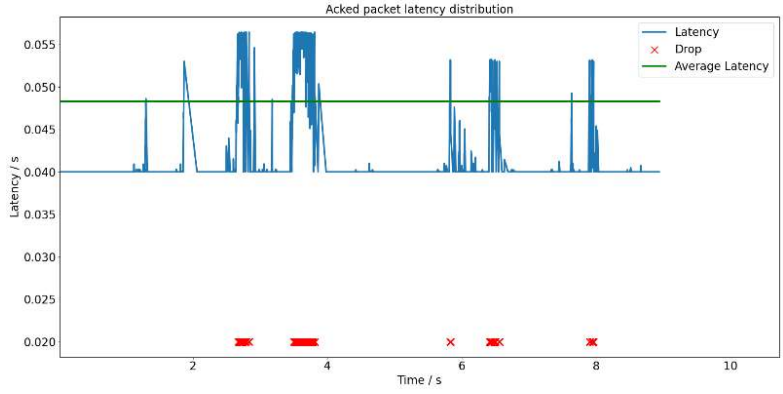

图5 Actor-Critic算法时延

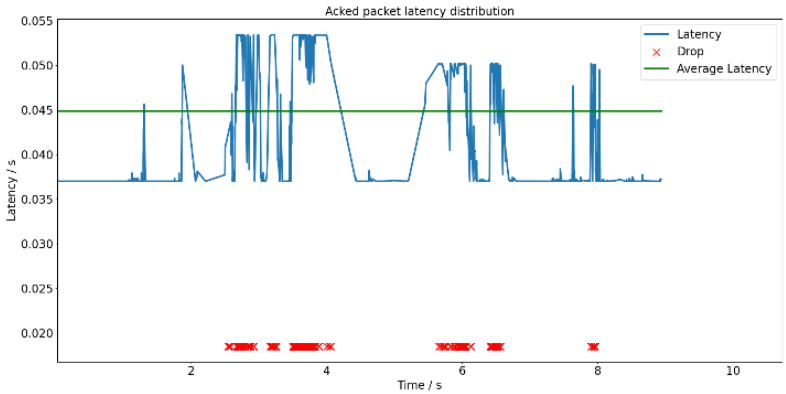

图6 本文算法时延

通过图7,本文展示了Reno算法与本文算法的窗口更改过程。可以看出,在丢包频繁的时间内,Reno算法的拥塞窗口出现较大波动,并出现了频繁的波动现象。而本文算法不仅考虑了丢包事件,还综合考虑了其他网络状态和特征,在丢包事件发生时拥塞窗口的改变更加稳定,能够更快地适应带宽的改变,并且能够更高效地利用网络带宽。

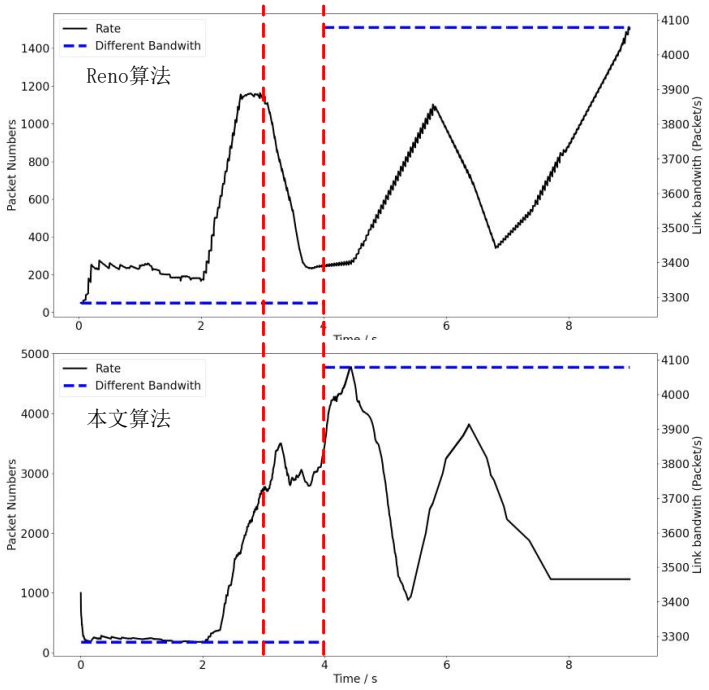

图7 Reno算法与本文算法拥塞窗口更改过程

5 结论

本文设计了一种基于DQN的延迟敏感业务资源调度算法:首先,本文说明了DQN用于延迟敏感业务的优势,基于强化学习对拥塞控制实现自适应改变,使得算法有能力进行动态的持续学习。其次,本文对比了在相同的排队算法条件下,Reno算法、Actor-Critic算法与本文所提算法在传输不同优先级数据包的延迟过程,比较了在不同网络状态下本文算法与Reno算法、Actor-Critic算法窗口更改过程。仿真结果表明,本文算法在动态丢包策略与时延上优于Reno算法、Actor-Critic算法,本文算法在网络发生频繁变化时窗口改变较Reno算法更稳定,最终本文算法用户体验质量远优于Reno算法。

参考文献(References):

[1] 金亚琪. 多业务QoS保障的资源分配算法研究[D]. 北京:北京邮电大学, 2018.

［2］ Wang Q，Chen M，Song X，et al. IGSO for QoS awareness web service composition optimization［C］// 2020 International Conference on Robots & Intelligent System（ICRIS），2020：509-512.

［3］ 金肱羽，武霄泳，张志龙，等. 面向视频流媒体直播的码率自适应算法研究［J］. 中国传媒大学学报(自然科学版)，2022，29(01)：8-14.

［4］ 廖晓闽，严少虎，石嘉，等. 基于深度强化学习的蜂窝网资源分配算法［J］. 通信学报，2019，40(2)：11-18.

［5］ Chmieliauskas D，Mahmood A，Thar K，et al. Q-Learning inspired method for antenna azimuth selection in cellular networks［C］// 2023 Workshop on Microwave Theory and Technology in Wireless Communications（MTTW），2023：7-12.

［6］ Burhanuddin L A B，Liu X，Deng Y，et al. QoE optimization for live video streaming in UAV-to-UAV communications via deep reinforcement learning［J］. IEEE Transactions on Vehicular Technology，2022，71(5)：5358-5370.

［7］ Wang Q，Wang Q，Zhao H，et al. Device-specific QoE enhancement through joint communication and computation resource scheduling in edge-assisted IoT systems［J］. IEEE Internet of Things Journal，2023：1-1.

［8］ Moura H D，Oliveira J M，Soares D，et al. Improved video QoE in wireless networks using deep reinforcement learning［C］// 2023 19th International Conference on Network and Service Management（CNSM），2023：1-7.

［9］ Saeedinia R，Fatemi S O，Lorenzi D，et al. Community-based QoE enhancement for user-generated content live streaming［C］// 2023 13th International Conference on Computer and Knowledge Engineering（ICCKE），2023：060-066.

［10］ Farahani R，Çetinkaya E，Timmerer C，et al. ALIVE：a latency-and cost-aware hybrid P2P-CDN framework for live video streaming［J］. IEEE Transactions on Network and Service Management，2023：1-1.

［11］ Sharmin Z，Roy P，Razzaque M A. User quality of experience and profit aware task allocation in mobile device cloud［C］// 2023 International Symposium on Networks，Computers and Communications（ISNCC），2023：1-6.

［12］ Kebedew T M，Ha V N，Lagunas E，et al. QoE-oriented resource allocation design coping with time-varying demands in wireless communication networks［C］// 2022 IEEE 96th Vehicular Technology Conference（VTC2022-Fall），2022：1-5.

［13］ Comşa I S，Muntean G M，Trestian R. An innovative machine-learning-based scheduling solution for improving live UHD video streaming quality in highly dynamic network environments［J］. IEEE Transactions on Broadcasting，2021，67(1)：212-224.

编辑：赵志军