分类号：TP391　　学校代码：10363
密　级：公开　　学　　号：2210920114

安徽工程大学
Anhui Polytechnic University

硕士学位论文

题目 无线环境下QoE感知的自适应比特率算法研究

论 文 作 者　刘江
校 内 导 师　李钧 副教授
校 外 导 师　刘冬 正高级工程师
专业学位类别　计算机技术
研究方向（领域）　智能系统与应用

论文提交日期：2024 年 5 月 23 日

分类号：TP391　　单位代码：10363
密 级：公开　　学　号：2210920114

题　目　**无线环境下 QoE 感知的自适应比特率算法研究**

英文并列题　目　**Research on Adaptive Bitrate Algorithms for QoE Perception in Wireless Environments**

论 文 作 者：刘江
校 内 导 师：李钧 副教授
校 外 导 师：刘冬 正高级工程师
专 业 学 位 类 别：计算机技术
研究方向（领域）：智能系统与应用

论文答辩日期：2024 年 5 月 24 日

安徽工程大学学位论文原创性声明

本人郑重声明：我恪守学术道德，崇尚严谨学风。所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已明确注明和引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品及成果的内容。论文为本人亲自撰写，我对所写的内容负责，并完全意识到本声明的法律后果由本人承担。

学位论文作者签名：刘江

日期：2024年5月23日

安徽工程大学学位论文版权使用授权书

学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅或借阅。本人授权安徽工程大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保密 □，在_____年解密后适用本版权书。

本学位论文属于

不保密 ☑。

学位论文作者签名：刘江　　指导教师签名：

日期：2024 年 5 月 23 日　　日期：2024 年 5 月 23 日

无线环境下 QoE 感知的自适应比特率算法研究

摘 要

自适应比特率(Adaptive Bitrate，ABR)算法部署在 DASH 客户端，根据当前网络状况、客户端缓存状态和本地硬件处理能力等因素，动态选择每个视频切片的比特率，其目的是最大化用户的观看体验质量(Quality of Experience，QoE)。它作为 DASH 技术的关键组成部分，一直备受研究者的关注。随着移动通信技术的高速发展以及无线终端的广泛普及，视频应用的无线化越发普遍。与有线网络环境相比，ABR 算法在无线环境下面临的挑战更多，具体而言体现在：1.无线网络带宽波动具有随机性和时变性，这导致现有的吞吐量预测器在无线网络中无法进行准确的吞吐量预测，从而影响 ABR 算法的比特率决策，间接影响用户 QoE。2.剧烈且频繁波动的无线网络带宽会导致现有的 ABR 算法频繁切换比特率，在带宽较高时选择高比特率视频，在低带宽较时选择低比特率视频，这导致视频质量和视频平滑度较低。3.无线网络环境下容易出现多客户端竞争同一网络带宽资源的现象，这导致现有的 ABR 算法在进行比特率决策时带宽分配不均匀，部分用户以低比特率播放，甚至卡顿，而少部分用户却能以高比特率播放。低视频质量、低视频平滑度和不公平的带宽分配都会严重影响用户的 QoE。因此针对无线环境下的 ABR 算法研究具有重要的理论意义和应用价值。

本文首先介绍了流媒体传输技术的发展和 DASH 技术的主要内容。然后，基于现有 ABR 算法的研究现状以及 ABR 算法在无线环境下面临的挑战，开展了针对无线环境下的 ABR 算法研究，本文的研究主要围绕着以下三个方面进行展开：

(1)针对现有的吞吐量预测器在无线环境下无法进行准确的吞吐量预测问题，设计并实现了一种基于 Informer 的吞吐量预测器 TPMI(throughput prediction model base on Informer)。与基于 HM、 LSTM 和 MLR 的预测器相比，TPMI 具有更好的预测性能，预测准确率得到明显提升。将 TPMI 集成到 MPC 算法中，与 Pensieve 和 BBA 算法相比，用户的 QoE 得到有效提升。

(2)针对在无线网络带宽剧烈且频繁波动时，现有的 ABR 算法选择的视频平均比特率低且比特率切换频繁，从而导致较低的视频质量和平滑度，提出了一种基于 QoE 感知的替换补偿算法。算法的核心思想是在带宽改善时,以更高比特率的视频切片动态替换缓冲区中尚未播放的低比特率视频切片。仿真实验表明，融合了该算法的 BOLA-E 算法和 THROUGHPUT 算法性能得到有效提升。

(3)针对现有的 ABR 算法在多客户端竞争同一网络带宽资源时，带宽分配不均匀且用户 QoE 较低的问题。提出了一种基于 COMA 的多客户端流媒体 QoE 优化算法。该算法基于集中式训练和分布式执行(CTDE)的多智能体强化学习框架,对多客户端比特率决策问题进行分布式部分可观测马尔可夫模型(Decentralized Partially Observable Markov Decision Process ,Dec-POMDP)建模，并通过基于反事实多智能体 COMA 策略梯度算法进行求解。仿真实验表明，与 FESTIVE、SFTM 等算法相比，视频卡顿时间明显减少，并且在维护各用户之间公平性的基础上有效提升了用户的 QoE。

关键词：无线，自适应比特率，吞吐量，QoE，DASH

RESEARCH ON ADAPTIVE BITRATE ALGORITHMS FOR QUALITY OF EXPERIENCE PERCEPTION IN WIRELESS ENVIRONMENTS

ABSTRACT

The Adaptive Bitrate (ABR) algorithm, deployed on DASH clients, dynamically selects the bitrate for each video segment based on factors such as current network conditions, client cache status, and local hardware processing capabilities, aiming to maximize users' Quality of Experience (QoE) in watching videos. As a crucial component of DASH technology, the ABR algorithm has long been the focus of researchers.With the rapid development of mobile communication technology and the widespread popularity of wireless terminals, the wireless application of video has become increasingly common. Compared to wired network environments, ABR algorithms face more challenges in wireless settings, specifically: 1. The randomness and time-variability of wireless network bandwidth fluctuations, which lead to the inability of existing throughput predictors to accurately forecast throughput in wireless networks, thereby affecting the bitrate decisions of ABR algorithms and indirectly impacting user QoE. 2. The severe and frequent fluctuations in wireless network bandwidth cause existing ABR algorithms to frequently switch bitrates, selecting high bitrate videos when bandwidth is high and low bitrate videos when bandwidth is low, resulting in lower video quality and smoothness. 3. The phenomenon of multiple clients competing for the same network bandwidth resources in wireless network environments, leading to uneven bandwidth distribution by existing ABR algorithms, with some users experiencing low bitrate playback or even buffering, while a few users are able to play at high bitrates. Low video quality, low smoothness, and unfair bandwidth distribution can severely affect user QoE. Therefore, research on ABR algorithms in wireless

environments has significant theoretical and practical value.

This dissertation first introduces the development of streaming media transmission technology and the main content of DASH technology. Then, based on the current state of research on ABR algorithms and the challenges faced by ABR algorithms in wireless environments, research on ABR algorithms in wireless environments is conducted, focusing on the following three aspects:

(1) In response to the issue of existing throughput predictors being unable to accurately forecast throughput in wireless environments, a throughput prediction model based on Informer (TPMI) was designed and implemented. Compared to predictors based on HM, LSTM, and MLR, TPMI exhibits superior predictive performance, with a significant improvement in prediction accuracy. Integrating TPMI into the MPC algorithm, compared to the Pensieve and BBA algorithms, effectively enhances user QoE.

(2) In response to the issue of existing ABR algorithms selecting videos with a low average bitrate and frequent bitrate switching due to severe and frequent fluctuations in wireless network bandwidth, resulting in lower video quality and smoothness, a QoE-aware substitution compensation algorithm was proposed. The core idea of the algorithm is to dynamically replace low bitrate video segments in the buffer that have not yet been played with higher bitrate segments when bandwidth improves. Simulation experiments show that the performance of the BOLA-E and THROUGHPUT algorithms, which incorporate this algorithm, is effectively enhanced.

(3) In response to the issue of existing ABR algorithms resulting in uneven bandwidth distribution and lower user QoE when multiple clients compete for the same network bandwidth resources, a multi-client streaming media QoE optimization algorithm based on COMA was proposed. This algorithm is based on a multi-agent reinforcement learning framework with centralized training and distributed execution (CTDE), models the multi-client bitrate decision problem as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP), and solves it using a counterfactual multi-agent COMA policy gradient algorithm. Simulation experiments show that, compared to the FESTIVE and SFTM algorithms, video buffering time is

significantly reduced, and user QoE is effectively enhanced while maintaining fairness among users.

KEY WORDS: wireless, ABR, throughput, QoE, DASH

目录

第 1 章 绪论

1.1 研究背景及意义

近年来，由于移动通信技术的快速进展以及无线终端设备在全球范围内的广泛普及，移动视频流量显著增长。文献[1]指出，截至 2022 年，视频流量将占到所有消费者流量和内容传输网络流量的 82%，其中 2017-2022 年数据显示，视频流量占所有移动数据流量的 79%。为了满足用户对移动视频业务的体验质量需求，技术层面的研究也在不断深入，主要分为移动智能终端发展和视频传输技术的提升。

流媒体传输技术是目前视频传输的主流技术，它将视频内容进行编码并存储在视频服务器中，然后通过互联网分段发送数据，以供用户观看[2]。与下载式相比，流式传输的优点在于，它不用等待完整视频内容下载完成之后才能观看，对用户缓存容量要求低。传统的流媒体传输方式包括基于 HTTP 的顺序流式传输和实时流式传输[3]。基于 HTTP 的顺序流式传输将编码好的视频文件按照先后次序传输到客户端，可以穿透常规设置的防火墙，可以立马观看已经下载的视频内容[4]。实时流式传输通过匹配网络连接带宽进行实时的传输数据，支持随机访问，用户可以进行实时的观看，能够适用于点播和直播[5]。虽然他们在曾经的一段时间里被广泛运用，但其缺点明显。基于 HTTP 的顺序流式传输不支持随机访问，只适用于点播，当网络带宽较低时，时延较长且容易发生卡顿[6]。实时流式传输对网络环境要求较高，需要特定的服务器和网络协议，并且在部署时容易受到防火墙拦截[7]。

因此，传统的流媒体技术逐渐被 DASH 技术所取代。在 DASH 传输系统的服务端，视频内容被编码为多个码率的版本，以应对不同的网络条件[8]。这些不同码率的视频进一步被切割为数秒长度的片段，且这些片段在时间序列上进行了精准对齐，以便客户端根据网络状况平滑切换比特率[9]。与视频切片同时生成的还有流媒体描述文件(Media Presentation Description, MPD)，它记录了每个视频切片的持续时长、存在的比特率种类和视频切片的 URL 等重要信息[10]。媒体客户端在接收到 MPD 之后，会利用 ABR 算法来动态选择最适合的视频比

特率，并下载相应的视频切片进行播放。

ABR 算法是 DASH 技术的主要研究热点，算法部署在用户客户端，它根据当前的网络状况、客户端缓存状态和客户端本地硬件的处理能力等因素，动态地选择每个视频切片的比特率，目的是最大化用户的观看体验质量(Quality of Experience，QoE)[11]。用户 QoE 的影响因素主要包括：视频的启动时间、视频的平滑度(比特率切换幅度和次数)、视频的卡顿(重新缓冲次数和重新缓冲时间)、视频的质量(视频比特率)、视频的内容等[12]。通常情况下，视频的质量和平滑度越高，用户 QoE 越大。相反用户 QoE 越小。同时卡顿和较长的启动时间(特殊的卡顿)也会严重降低用户的 QoE。

无线网络技术的发展为移动流媒体技术奠定了重要基础，然而也为 ABR 算法的研究带来了严峻的挑战，具体而言主要体现在：(1)无线网络带宽波动具有随机性和时变性，这导致现有的吞吐量预测器在无线环境下无法准确的预测，从而导致现有 ABR 算法无法及时响应带宽的变化，从而影响比特率决策。主要表现形式为：在网络带宽上升时，依旧选择低比特率视频，从而造成了带宽资源的浪费。在网络带宽下降时，仍然维持原有高比特率视频的选择，加剧了视频卡顿。低比特率视频和卡顿都会严重影响用户的 QoE。(2)剧烈且频繁波动的无线网络带宽会导致现有 ABR 算法在视频传输过程中频繁切换视频比特率，在带宽较高时选择高比特率视频，在低带宽较时选择低比特率视频，这导致视频质量和视频平滑度较低。低视频质量和低视频平滑度都会严重影响用户的 QoE。(3)无线网络环境下容易出现多客户端竞争同一网络带宽资源的现象，导致各客户端在选择比特率时的决策会相互影响。当所有用户都选择高比特率视频时，可能会引起网络拥堵，进而造成播放时的延迟或卡顿。而为了避免这种情况，降低部分用户的比特率又会导致用户间的 QoE 产生较大的差异，因此带宽资源分配的公平性至关重要。

低视频质量、低视频平滑度和视频卡顿都会严重影响用户的 QoE。因此，为了更好的满足用户在无线环境下的 QoE 需求，开展针对无线网络环境下 ABR 算法的研究，对于实际应用和理论探索都具有极高的意义。

1.2 国内外研究现状

现有的 ABR 算法主要分为：基于缓存的方法、基于带宽的方法、基于带宽和缓存因素的混合方法以及基于学习的方法。

1.2.1 基于带宽的 ABR 算法

基于带宽的 ABR 算法是客户端根据当前的网络带宽，选择合适的比特率，该方法主要包括可用带宽估计和比特率选择两个重要组成部分。现有的方法主要有：

Liu 等人在文献[13]中提出了一种 ABR 算法，该算法尝试使用基于段获取时间(segment fetch time ，SFT)的平滑网络吞吐量来检测带宽波动和拥塞，SFT 测量从发送 HTTP GET 请求到接收段最后一个字节的时间。作者在文献[14]中扩展了他们的工作，通过使用比较预期段获取时间(ESFT)和测量的 SFT 的度量来确定所选段比特率是否与网络容量匹配。Rainer 等人在文献[15]中采用了类似的方法，根据最后一个下载段观察到的比特率和前一次估计期间计算的估计吞吐量来计算下一个段的估计带宽。初始化是基于下载 MPD 时测量的带宽。

Li 等人提出的 Panda 算法[16]准确地估计可用带宽，并试图消除 ON-OFF 稳态问题，并减少当多个客户端共享同一瓶颈链路时的比特率振荡。用于 LTE 网络中的 DASH 客户端的视频适应框架。FESTIVE[17]算法通过实时监测网络状况并动态调整视频流的比特率，以确保在多用户环境下视频播放的公平性、效率和稳定性。

文献[18]中提出了一种 piStream 方法，它使客户端能够基于充当物理层守护进程的资源监控模块来估计可用带宽。Miller 等人在文献[19]中提出了一种基于低延迟预测的无线接入链路自适应比特率方案，称为基于低延迟预测的自适应(LOLYPOP)，该方案利用 TCP 在多个时间尺度(即 1 到 10 秒)上的吞吐量预测来实现低延迟并提高查看器 QoE。

基于带宽的 ABR 算法主要是建立在带宽估计的基础上，带宽估计的准确性是该类方法面临的挑战。带宽估计不准确会导致比特率波动频繁，甚至发生卡顿。

1.2.2 基于缓存的 ABR 算法

基于带宽的 ABR 算法侧重于对当前缓存状态的持续监测，并依据不同缓存状态采取相应自适应比特率策略。现有方法主要包括：

Sieber 等人提出了一种基于 SVC(Scalable Video Coding)的自适应算法 BIEB[20]。BIEB 算法利用 SVC 的分层编码特性，通过优先选择关键层的视频数据，在最大化视频质量的同时，减少了质量振荡的次数，有效避免了视频播放的卡顿和频繁的比特率切换。BIEB 算法的核心思想是在提高视频质量(即增强层)之前，先保证稳定的缓冲区占用。通过维持一个合适的缓冲区水平，BIEB 算法能够应对网络带宽的波动，提供更加平滑和连续的视频播放体验。这种缓冲区管理策略，有效地平衡了视频质量和播放稳定性之间的关系，提高了用户的 QoE。然而，BIEB 算法在设计时没有充分考虑网络中动态交叉流量的影响。当网络中存在其他竞争流量时，可用带宽可能会发生剧烈变化，导致视频传输的比特率切换或卡顿现象。BIEB 算法虽然通过缓冲区管理策略减少了比特率切换的频率，但在动态交叉流量的情况下，仍然可能出现视频质量下降或卡顿的问题。

Huang 等人提出了一种基于缓冲区的 ABR 算法，称为 BBA(Buffer-Based Adaptation)[21]。BBA 算法的主要目标是最大化视频的平均质量，同时避免不必要的再缓冲事件，以提供更加流畅和稳定的视频播放体验。BBA 算法通过监测客户端的缓冲区状态，动态调整视频的比特率。当缓冲区占用较高时，算法倾向于选择较高的比特率，以提高视频质量；当缓冲区占用较低时，算法则选择较低的比特率，以避免再缓冲事件的发生。这种基于缓冲区的自适应策略，能够有效地平衡视频质量和播放连续性之间的关系，提高了用户的 QoE。在长期带宽波动的情况下，BBA 算法可能会频繁地调整视频比特率，导致视频质量的不稳定和用户体验的下降。频繁的比特率切换会引起视觉上的不连续性，影响用户的观看体验。此外，当网络带宽持续低于视频比特率时，BBA 算法可能会导致缓冲区不断消耗，最终引发再缓冲事件，进一步降低用户的 QoE。

Mueller 等人在文献[22]中针对基于带宽的自适应算法在多客户端竞争可用带宽时的局限性，特别是在存在缓存服务器的情况下，提出了一种新的自适应比特率方案。该算法通过考虑缓冲区大小，实现了更加准确的比特率选择和平

滑的比特率切换，有效地提高了视频传输的质量和用户体验，为解决多客户端竞争带宽的问题提供了新的思路。

Spiteri 等人于提出了 BOLA[23]算法，该算法基于 Lyapunov 方法，将自适应比特率问题视为一个在线控制算法，以最大化效用函数。该效用函数综合考虑了平均比特率和重新缓冲时间，同时适应网络状况的变化，以获得更好的 QoE。作者提供了严谨的理论证明，表明 BOLA 算法的性能接近最优。此外，他们设计了一个包含平均播放质量和再缓冲时间的 QoE 模型，并通过各种网络轨迹的实验验证了 BOLA 算法的效率。作为一种基于缓冲区的算法，BOLA 已在 dash.js 播放器中实现并可供使用。

Yadav 等人提出了一种基于排队理论的 DASH 自适应比特率方法 QUETRA[24]。他们将 DASH 客户端建模为 M/D/1/K 队列，通过该模型可以计算给定比特率选择、网络吞吐量和缓冲区容量下的预期缓冲区占用。基于这个模型，作者提出了一种简单的 ABR 算法。尽管 QUETRA 算法较为简单，但在多种场景下的评估结果表明，它相比现有的算法能够产生更好的 QoE。

基于缓存的自适应算法虽然充分考虑了缓存但没有考虑带宽情况，即使当网络带宽足够好时，在 ABR 算法的启动阶段，因为缓冲区占比低，会选择下载低比特率视频切片，从而造成带宽资源的浪费。并且基于缓存的 ABR 算法对带宽变化不能及时响应，导致在带宽突然下降时，仍维持原来相对较高的比特率选择，从而造成视频卡顿。

1.2.3 基于混合的 ABR 算法

基于带宽和缓存因素混合的 ABR 算法是指在选择不同比特率切片时，同时考虑带宽和缓存因素的影响。现有方法主要有：

Miller 等人在文献[25]中提出了一种 ABR 算法。该算法使用当前缓冲区占用水平、估计可用带宽以及媒体展示描述(MPD)中不同比特率水平的平均比特率作为比特率选择的指标。该算法旨在(i)准确估计可用带宽,避免带宽高估;(ii)在最小化启动延迟、失速次数、质量振荡和播放中断的同时,最大化比特率。算法根据当前缓冲区水平调整其行为,可以提高竞争客户端之间的公平性。然而,该算法没有考虑任何与观众满意度相关的指标。此外,在共享网络环境中,即使客户端达到最高质量水平,也可能遭受视频不稳定、失速和质量波动的困扰。这是由

于在比特率决策过程中缺乏对观众体验质量(QoE)的考虑。

L. De Cicco 等人提出了一种名为 ELASTIC[26]的方法，该方法基于反馈控制理论，是一种自适应流媒体控制器。ELASTIC 通过生成持久的 TCP 流，避免了导致带宽过估的 ON-OFF 稳态行为。引入 ELASTIC 的目的是确保基于网络反馈的竞争客户端之间的带宽公平性，但它并未考虑观看者的 QoE（Quality of Experience，用户体验质量）。此外，ELASTIC 在比特率决策过程中忽略了视频质量波动这一因素。因此，无论是在带宽波动的情况下，还是在固定带宽环境中，ELASTIC 都可能导致频繁的比特率切换，从而影响用户的 QoE。尽管 ELASTIC 在确保客户端之间的带宽公平性方面有所贡献，但其未能全面考虑用户体验质量，在实际应用中可能会导致较差的 QoE 表现。

Yin 等人在文献[27]中提出了一个控制理论框架，该框架有助于理解和探究在面对不同网络带宽变化时，基于带宽和基于缓冲区的自适应算法之间的权衡取舍。基于该框架，作者设计了一种实用的模型预测控制器 MPC。该控制器能够最优地结合带宽和缓冲区大小的预测信息，从而为下一视频段选择最合适的比特率，以实现最大化用户的 QoE。J. M. Batalla 等人提出了一种名为 SARA（Segment-Aware Rate Adaptation）[28]的算法，该算法是一种基于视频段大小变化、可用带宽估计和缓冲区占用情况进行自适应调整的段感知带宽自适应算法。鉴于 HTTP 协议使用 TCP 作为传输层协议，视频段的吞吐量与文件大小密切相关。因此，作者建议对典型的 MPD（Media Presentation Description，媒体展示描述）文件进行增强，使其包含每个视频段的大小信息。在每次下载新的视频段时，客户端根据使用段大小评估得到的估计带宽以及当前缓冲区的状态，来决定下一个视频段的质量级别。该算法充分考虑了视频段大小、网络带宽和缓冲区占用等因素，能够实现更加精确和高效的自适应比特率调整。

C. Wang 等人提出了一种基于频谱的 ABR 算法 SQUAD[29]。SQUAD 利用可用带宽和缓冲区信息来提高视频的平均比特率，同时最大限度地减少质量切换的次数。在启动阶段，SQUAD 采用保守的方法获取更多低质量的视频段，以减轻未来带宽估计中可能由单个低质量段估计引起的任何不准确性。接下来，该算法综合考虑频谱(即平均段比特率的变化)和缓冲区水平，来选择下一个视频段的比特率。SQUAD 算法的优点在于，它能够根据网络状况和缓冲区水平动态

调整视频的比特率，从而在保证视频质量的同时，尽可能减少质量切换的次数。这种自适应的比特率调整策略，有效地提高了视频传输的效率和用户的观看体验。

Sobhani 等人在文献[30]提出了一种基于模糊逻辑机制的 ABR 算法。该算法通过预测可用带宽和缓冲区水平，来选择合适的视频比特率。模糊逻辑机制的引入，使得算法能够更好地处理网络状况的不确定性和复杂性，提高了比特率选择的准确性和适应性。然而，该算法在多个客户端竞争可用带宽的情况下，仍然存在一些局限性。当多个客户端同时请求视频内容时，算法只能确保每个客户端获得一致的视频质量，而没有考虑客户端之间的公平性问题。这可能导致某些客户端获得较高的视频质量，而其他客户端却只能获得较低的视频质量，从而影响用户的观看体验。

Bentaleb 等人提出了一种 GTA 算法[31]。该算法基于博弈论框架，将比特率选择问题表述为一个议价过程和共识机制。在该方案中，多个 DASH 客户端通过形成联盟的方式进行合作博弈。每个客户端作为博弈中的一个玩家，通过与其他客户端协商和达成共识，来决定各自的比特率选择策略。在 GTA 方案中，客户端之间的协商过程遵循一定的议价规则和共识机制。通过这种协商和共识过程，客户端可以在满足各自 QoE 目标的同时，实现公平、高效的比特率分配。这种基于博弈论的方法，能够有效地解决多客户端竞争带宽资源时的冲突和不公平问题，提高了整个系统的性能和用户体验。然而 GTA 算法需要客户端之间频繁地交换信息，以达成共识和协调比特率选择，这在实际部署中式难以实现的。

综上考虑混合因素的 ABR 算法重点在带宽利用和缓存利用之间取得更佳均衡的表现。

1.2.4 基于学习的 ABR 算法

在基于学习的 ABR 算法中，视频流比特率决策过程被表述为一个有限的马尔可夫决策过程，以便能够在波动的网络条件下做出自适应决策。现有的算法主要有：

Xing 等人[32]提出了一种基于多址网络的实时最优动作搜索算法,旨在提供流畅、高质量的视频播放体验。该算法利用蓝牙和 WiFi 连接同时下载视频切片。

在每个状态下,作者构建了一个马尔可夫决策过程(MDP)模型,其中带宽适配代理将缓冲区水平、可伸缩视频编码(SVC)层索引、蓝牙流量、可用带宽以及获取的每个视频切片的索引作为输入。奖励函数的设计综合考虑了平均播放质量、播放中断率和播放平滑度等因素。然而,该方案在用户移动场景下表现出一定的局限性,这可能对用户的 QoE 产生负面影响。

Bokani 等人在文献[33]中提出了一种基于马尔可夫决策过程(MDP)和强化学习(RL)的方法,用于优化车载环境下的 HTTP 自适应流媒体传输。该方法将自适应比特率问题建模为一个顺序决策问题,并使用 Q-learning、Softmax Q-learning 和 Dueling Q-learning 三种 RL 算法求解 MDP,实现比特率的动态调整。同时,利用真实车载网络测量数据构建带宽估计模型,提高算法的性能。但算法复杂度较高,实际部署可能存在挑战，实验评估缺乏真实环境验证，未考虑算法的计算开销和时延影响。

Zhou 等人提出 mDASH 算法[34],该算法基于马尔可夫决策过程(MDP)框架。它将自适应比特率问题建模为一个 MDP 问题,状态包括当前缓冲区占用量和上一个视频段的比特率,动作为选择下一个视频段的比特率,奖励函数考虑了缓冲区占用量、比特率切换频率和视频质量等因素。通过动态规划算法求解 MDP 问题,得到最优的比特率选择策略。算法通过在线学习估计 MDP 模型参数,并使用滑动窗口方法更新模型,以适应网络条件的变化。该算法具有自适应能力，能在一定程度上提高视频质量,减少卡顿。但 MDP 建模和求解复杂度较高,实时性可能受限，奖励函数设计依赖于经验,泛化性较差。

Mao 等人提出的 Pensieve[35]和 Gadaleta M 等人提出的深度 Q 学习 DASH(D-DASH)[36]方法,都是利用深度强化学习(Deep RL)技术来提高比特率决策估计的准确性和速度。Pensieve 是一个基于 DASH 客户端收集的观察结果(即吞吐量估计和缓冲区占用)构建的框架,并在大规模视频流实验中进行了测试。与依赖预编程模型或对环境做出假设的方法不同,Pensieve 通过观察和经验逐步学习最优的比特率决策策略。D-DASH 则结合了深度学习和强化学习机制,以改善 DASH 的用户体验质量(QoE),并在决策过程中实现了策略最优性和收敛速度之间的良好平衡。具体而言,D-DASH 采用了包括前馈和循环深度神经网络在内的混合学习架构和高级策略。这两种解决方案都展现出优异的性能,证明了将深度强化学习

与自适应比特率(ABR)启发式方法相结合在比特率决策过程中的优势。所提出的基于马尔可夫决策过程(MDP)的方案在减少视频卡顿方面显著提升了整体性能,从而确保了可接受的用户观看体验质量。然而,当客户端数量增加时,这些方案可能会面临不稳定、不公平和资源利用率低下等问题。这可能是由于 MDP 模型没有考虑这些因素,以及客户端去中心化的 ON-OFF 模式导致的。

HotDASH[37]算法利用深度强化学习(Deep RL)技术,实现了视频热点片段的适时预取,以提高用户的 QoE。该算法在 DASH.js 播放器中实现了一个预取模块,该模块由一个最优预取和比特率决策引擎驱动。决策引擎被设计为一个级联的强化学习模型,使用基于神经网络的 Actor-Critic 算法实现。通过在各种带宽条件下进行基于轨迹的模拟,对神经网络进行训练。HotDASH 能够根据用户偏好进行适时预取,提高了用户 QoE。但算法复杂度较高,实际部署可能面临挑战;预取策略的有效性依赖于用户偏好的准确预测。

Tianchi Huang 等人提出了 Comyco 算法[38],这是一种基于深度强化学习的自适应比特率(ABR)算法。该算法通过离线训练深度神经网络模型,学习网络状态到最优比特率的映射策略。在线推理阶段,Comyco 实时提取网络状态特征,并使用训练好的策略网络做出比特率决策。与传统的 ABR 算法相比,Comyco 在保证用户体验质量(QoE)方面表现出色,同时推理开销较小,适合应用于资源受限的场景。然而,Comyco 算法也存在一些局限性。首先,离线训练过程依赖于大量的数据,这可能在实际应用中难以获取或者需要较高的时间和计算成本。其次,Comyco 主要关注单个用户的 QoE 优化,未充分考虑多用户竞争网络资源的情况。

Yashuang Guo 等人在文献[39]中提出了一种结合移动边缘计算（MEC）的自适应比特率（ABR）流媒体传输框架，通过在无线接入网络(RAN)边缘进行视频转码和质量适配,解决了传统 ABR 流媒体在无线网络中面临的拥塞和高延迟问题。将无线信道建模为有限状态马尔可夫信道,并将问题表述为一个随机优化问题,旨在最大化用户感知的 QoE 与边缘服务器转码成本之间的权衡。通过使用深度强化学习(DRL)算法,提出了一种自动执行计算资源分配和视频质量适配的算法,无需任何先验的信道统计信息。该算法有效解决无线网络中的拥塞和延迟问题，但未考虑多用户场景下的资源竞争和公平性问题。

Kevin Gatimu 等提出了一种 qMDP 算法[40]，该算法通过结合马尔可夫决策过程（MDP）和 M/D/1/K 队列部分建模的强化学习（RL）方法，旨在提高消费者的体验质量（QoE）。与仅基于 QoE 的无模型版本相比，qMDP 不仅能够提供更高的 QoE，还能实现更快的收敛速度。这种方法通过智能化地模拟 ABR 流程，解决了传统 RL 方法由于其无模型特性而导致的高复杂性和长收敛时间的问题。但 qMDP 仍然依赖于 RL 训练,在某些情况下可能需要大量的数据和计算资源，M/D/1/K 队列模型可能无法完全捕获真实网络环境的动态特性。

Weihe Li 等提出了一种基于 DRL 的 ABR 算法 RAV[41],它同时考虑了音频和视频质量对用户体验的影响。与现有的仅关注视频比特率选择的 ABR 算法不同,RAV 通过协调音视频比特率的选择,避免了低播放质量、频繁卡顿、播放不流畅以及音视频组合不匹配等问题。RAV 训练了一个神经网络模型,无需对环境做出任何假设,即可自动输出未来音视频块的最优比特率,具有良好的鲁棒性。实验表明 RAV 显著提高了用户的平均观看体验质量,在主观实验中得到了用户的认可;算法具有良好的适应性和鲁棒性。但 RAV 的计算复杂度较高,可能需要更多的计算资源;算法的泛化能力有待进一步验证,特别是在不同类型的音视频内容和网络条件下。

Wenzhong Li 等提出了一种 MetaABR[42]算法，这是一种基于元强化学习的自适应比特率(ABR)选择算法,旨在最大化用户的 QoE。该算法通过共享的元评论家(meta-critic)对多个学习任务进行联合训练,提供可迁移的元知识来指导跨任务的比特率选择。在未知环境中，MetaABR 只需少量尝试即可高效地学习新任务。算法具有良好的泛化能力和快速适应新环境的能力。

基于学习的自适应比特率(ABR)算法利用深度强化学习和元学习技术来优化视频流媒体的体验质量(QoE)，展现了卓越的自适应性和个性化特性。然而，这些算法对计算资源的需求较高，训练和收敛过程耗时较长，并且在数据稀缺或网络条件剧烈变动的环境中面临实际应用和保持稳定性方面的挑战。

1.3 主要研究内容

在数字化和移动互联网的快速发展背景下，视频流媒体服务已变得至关重要，广泛应用于在线教育、远程工作、娱乐播放及实时新闻等领域。随着智能设备普及和对高质量视频的需求增长，无线网络的不稳定性，如带宽波动和用

户间竞争，严重影响了视频服务的质量和流畅性。这些挑战使得传统的自适应比特率（ABR）算法难以有效调整，影响了视频质量和播放体验。因此，业界和学术界正积极研发新技术和算法，以优化无线环境下的视频流服务，提升用户体验。

本文主要对无线环境下的 ABR 算法进行研究，通过分析现有 ABR 算法在无线网络环境面临的挑战，从不同角度出发，提出相应的解决方案。如图 1-1 所示，本文的主要研究内容包括：

（1）无线环境下吞吐量预测研究。针对现有的吞吐量预测器在无线环境下无法进行准确的吞吐量预测的问题，通过分析无线环境下影响吞吐量预测的因素，设计并实现了一种基于 Informer 的吞吐量预测器 TPMI，并将其集成到现有的 ABR 算法中，以提升 ABR 算法的性能。

（2）无线环境下 ABR 算法优化研究。针对在无线网络带宽剧烈且频繁波动时，现有的 ABR 算法选择的视频平均比特率低且比特率切换频繁，从而导致的视频质量和平滑度低的问题，提出了一种基于 QoE 感知的替换补偿算法。

（3）多客户端 ABR 算法研究。针对现有的 ABR 算法在多客户端竞争同一网络带宽资源时，带宽分配不均匀且用户 QoE 较低的问题。提出了一种基于 COMA 的多客户端流媒体 QoE 优化算法。

（4）对本文设计的吞吐量预测器和所提算法进行实验验证和结果分析。

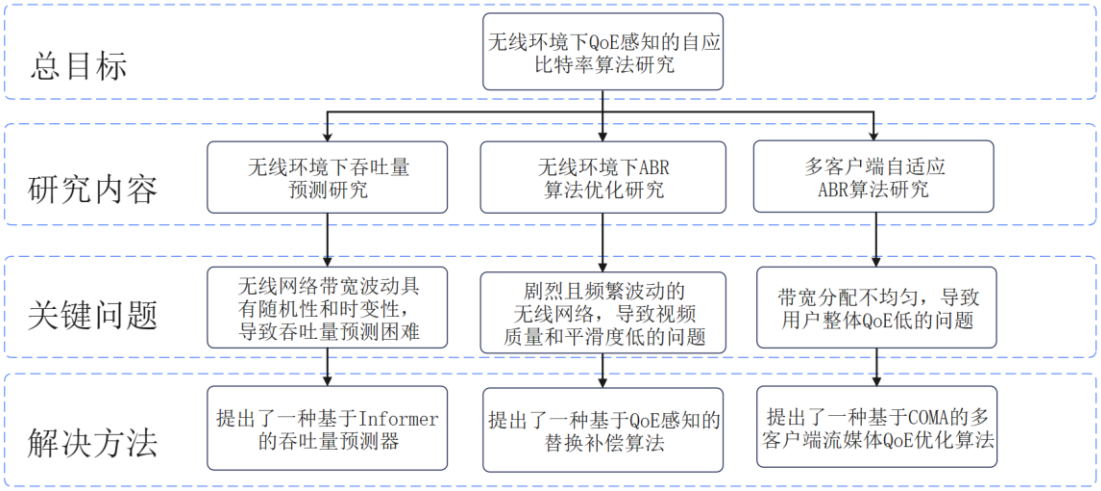


图 1-1 论文研究思路框架图

Figure 1-1 Framework diagram of the research idea of the dissertation

1.4 论文组织结构

第一章为绪论。主要介绍了本文的研究背景和意义以及现有 ABR 算法的研究现状。首先介绍流媒体技术的发展以及 ABR 算法的相关背景知识，然后描述 ABR 算法研究在无线环境中面临的问题，接着根据分类描述了现有 ABR 算法的研究现状。最后给出了本文的研究内容以及安排。

第二章为相关技术概述。主要介绍了流媒体传输技术以及 DASH 技术的主要内容。首先介绍了流媒体传输技术，其中包括传统流媒体传输技术以及基于 HTTP 的自适应流技术。接着介绍了 DASH 技术的主要内容，其中包括 DASH 技术架构、MDP 文件、视频切片、比特率控制算法以及 ABR 算法。最后介绍了强化学习理论和多智能体强化学习，其中包括 MDP、POMDP 以及多智能体强化学习框架。

第三章为吞吐量预测器 TPMI。本章首先进行了问题描述，指出现有吞吐量预测器的在无线环境下的不足。其次，阐述了本章的研究目标，设计一种适用于无线环境下的吞吐量预测器。然后介绍了 Informer 模型。随后设计并实现了一种基于 Informer 模型的吞吐量预测器。最后在仿真平台评估了该吞吐量预测器的预测性能，并且将它集成到现有的 ABR 算法中，以提升 ABR 算法的性能。

第四章为基于 QoE 感知的替换补偿算法。首先描述了无线流媒体 QoE 的影响因素。其次，进行了问题描述，主要为现有 ABR 算法在无线环境下的不足。然后阐述了本章的研究目的，优化现有 ABR 算法在无线环境下的性能。随后针对现有的 ABR 算法在无线网络带宽剧烈频繁波动时，视频质量和平滑度低的问题，提出了一种基于 QoE 感知的替换补偿算法。最后对算法进行了性能评估，并分析了实验结果。

第五章为基于 COMA 的多客户端流媒体 QoE 优化算法。本章首先对多客户端带宽竞争场景下的比特率决策进行问题描述。其次阐述了本章的研究目的，解决现有的 ABR 算法在多客户端竞争同一网络带宽资源时，带宽分配不均匀且用户 QoE 较低的问题。然后针对该问题提出了一种基于 COMA 的多客户端流媒体 QoE 优化算法。最后在模拟平台上对算法进行了评估，并对实验结果进行了分析。

第六章为总结与展望，描述了本文的主要工作以及未来的研究方向。

第 2 章 相关技术概述

2.1 流媒体传输技术

流媒体传输技术是一种通过网络实时传输音频、视频等多媒体内容的技术。它利用流式传输技术，将连续的影像和声音等多媒体内容经过压缩编码存储于服务器上，然后通过 HTTP、RTMP、RTSP 等协议将这些内容以流的形式传输至客户端，客户端经过解码实现观看和收听。与传统下载方式不同，在流媒体传输技术中心，用户可以边下载边观看和收听，无需等待整个文件下载完成。流媒体传输技术的优点在于实时性、适应性和带宽节省，因此在在线直播、视频点播和音乐播放等领域得到广泛应用。它主要分为传统的流媒体传输技术和基于 HTTP 的自适应流技术。

2.1.1 传统的流媒体传输技术

传统的流媒体传输技术包括基于 HTTP 的顺序流式传输和实时流式传输。基于 HTTP 的顺序流式传输将编码好的视频文件按照先后次序传输到客户端，可以穿透常规设置的防火墙，可以立马观看已经下载的视频内容。实时流式传输通过匹配网络连接带宽进行实时的传输数据，支持随机访问，用户可以进行实时的观看，能够适用于点播和直播。虽然他们在曾经的一段时间里被广泛运用，但其缺点明显。基于 HTTP 的顺序流式传输不支持随机访问，只适用于点播，当网络带宽较低时，时延较长且容易发生卡顿。实时流式传输对网络环境要求较高，需要特定的服务器和网络协议，并且在部署时容易受到防火墙拦截。

2.1.2 基于 HTTP 的自适应流媒体传输技术

基于 HTTP 的自适应流媒体传输技术是一种先进的流媒体传输方法，它通过允许客户端根据当前的网络条件和缓冲状况动态调整视频流的质量，从而优化用户体验。这种技术的出现，主要是为了解决传统基于 HTTP 的顺序流传输方式在带宽利用率低下和播放体验不佳等方面的不足。特别是在网络条件不稳定的情况下，如无线网络环境，HTTP 自适应流传输技术能够有效保证视频播放的流畅性，避免了视频播放过程中的卡顿现象。因此，基于 HTTP 的自适应流技术自提出以来备受广大视频服务商和研究者的关注。多个知名科技企业相继

推出了基于 HTTP 的自适应流技术，这些技术包括苹果公司的 HTTP 实时流媒体（HTTP Live Streaming，HLS）、微软公司的平滑流媒体（Smooth Streaming，SS）、以及 Adobe 公司的 HTTP 动态流媒体（HTTP Dynamic Streaming，HDS）。它们各自为流媒体传输提供了不同的解决方案，成功地促进了在线视频服务的发展，使得用户能够根据自己的网络条件享受到更加流畅的视频观看体验。

然而，随着这些技术的普及，它们之间的不兼容性问题开始浮现，这对媒体服务商和终端用户造成了困扰，限制了这些技术的更广泛应用。为了克服这一挑战，国际标准组织第三代合作伙伴项目（3GPP）在 2009 年首次提出了基于 HTTP 的动态自适应流（DASH）的概念，并通过与国际标准化组织 MPEG 的紧密合作，于 2012 年正式推出了 MPEG-DASH 标准[43]。这一标准旨在通过提供一个统一的 HTTP 自适应流传输框架，解决不同自适应流技术之间的兼容性问题，推动行业的标准化发展。

MPEG-DASH 作为一个国际认可的标准，为视频内容的组织和传输提供了清晰的指导，定义了媒体呈现描述（MPD）和媒体内容的格式规范。重要的是，该标准在确保统一的同时保留了足够的灵活性，允许内容提供商、服务商和设备制造商根据自己的具体需求和技术条件，对视频编解码、传输协议以及客户端的自适应比特率策略进行自定义和优化。这种可扩展性使得 MPEG-DASH 不仅能够满足当前的市场需求，同时也具备适应未来技术发展和新应用场景的能力。

2.2 MPEG-DASH 技术主要内容

2.2.1 MPEG-DASH 技术框架

DASH 技术的系统框架如图 2-1 所示，主要由 DASH 流媒体服务器和 DASH 客户端播放器组成。DASH 流媒体服务器存储着视频切片文件和 MPD 文件。其中 MDP 文件用来描述每个视频块持续时间、比特率等级个数、视频资源统一资源定位符(Uniform Resource Locator，URL)等信息。视频切片是视频内容被编码成多个具有不同质量级别的版本，每个版本根据比特率的不同提供了分辨率和数据量的变化。这些版本进一步被划分为若干时长固定的小段，即视频切片，以便根据网络条件动态调整播放质量。高比特率的视频切片意味着更高的画质和更大的文件大小，从而实现了在变化的网络环境中优化用户观看体验的目的。

在 DASH 流媒体技术中，客户端首先与 HTTP 服务器建立 TCP 连接，随后发出 HTTP-GET 请求以获取 MPD（媒体呈现描述）文件，该文件详细记录了视频的各种属性，如格式、分辨率、比特率、持续时间以及视频切片的 URL 地址。服务器接收到请求后，响应并传输 MPD 文件给客户端。客户端解析 MPD 文件后，根据当前的网络状况和缓存情况，采用自适应策略动态请求适宜比特率的视频切片，以优化网络带宽使用并确保视频播放的流畅性[44]。例如，在请求第 k 个视频切片时，如果网络状况良好并且缓存充足，客户端会选择请求较高比特率的切片。服务器随后发送所请求的视频切片至客户端，客户端进行解码播放。通过这一连续的过程——下载、解码播放视频切片同时调整比特率，DASH 技术确保了视频内容的平稳播放。

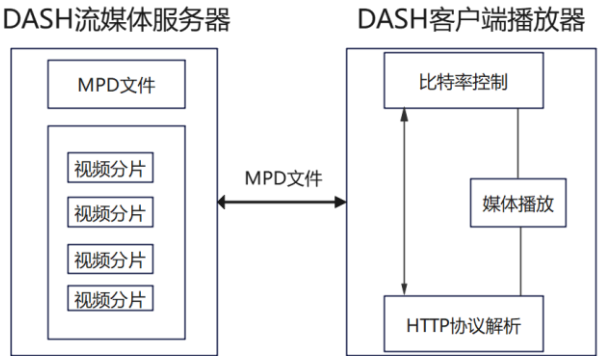


图 2-1 MPEG-DASH 技术系统框架

Figure 2-1 ,MPEG-DASH technology system framework

2.2.2 MDP 文件

在 DASH 流媒体传输协议中，MPD（媒体呈现描述）文件扮演着关键的角色，它是一种基于 XML 格式的文档，用于详细描述服务器上可用的媒体资源信息。通过解析 MPD 文件，客户端能够获取到包括视频切片的 URL 地址、各切片的比特率、分辨率、持续时间以及编码格式等重要信息。这些信息使得客户端能够根据当前网络条件和缓存情况，智能选择并请求适合的视频切片，进而实现媒体内容的高效下载和流畅播放。MPD 文件的设计充分利用了 XML 的易读性和机器可解析性，为 DASH 技术中的自适应流传输提供了强大的支持。

2.2.3 视频切片

在 DASH 流媒体技术中，视频内容的传输和播放依赖于几种关键类型的切片，主要包括初始化切片、媒体切片、索引切片和切换切片[43]。初始化切片主

要负责在视频播放开始时对播放器进行必要的设置，提供了播放所需的非媒体信息，为后续的视频播放奠定基础。媒体切片则携带实际的视频数据，是视频播放过程中不断请求和加载的主要内容，每个媒体切片通常包含一个或多个可以独立解码的码流入口点，使得解码器能够从任一入口点开始解码，无需依赖之前的数据。索引切片包含了媒体切片的索引信息，帮助播放器快速定位到特定的媒体内容。切换切片提供了从当前播放点切换到新的媒体切片所需的信息，支持播放过程中的流畅切换。这些切片类型共同构成了 DASH 流媒体传输的基础，使得客户端能够根据网络状况动态调整播放质量，确保用户体验的最优化。

2.2.4 比特率控制算法

比特率控制算法在视频传输过程中起着至关重要的作用。它决定了视频数据的编码带宽，以便在保持视频质量的同时，最大限度地减少数据传输的带宽需求。比特率控制算法的目标是在给定的网络条件下，提供最佳的视频质量，同时避免网络拥塞和视频播放中的缓冲。

比特率控制算法的发展历程可以追溯到 20 世纪 80 年代，当时的算法主要是基于场景复杂性的静态比特率分配。然而，这些方法在动态变化的网络环境中表现不佳。随着互联网的发展和视频流媒体技术的进步，比特率控制算法也发生了显著的变化。在 21 世纪初，研究人员开始开发出更为复杂的动态比特率控制算法，这些算法能够根据网络条件的变化动态调整视频的编码比特率。

近年来，ABR 算法已经成为了主流的比特率控制策略。目前，比特率控制算法的研究仍在不断发展中，研究人员正在探索如何利用机器学习和人工智能技术，进一步提升比特率控制算法的性能。

2.3 自适应比特率算法

自适应比特率（ABR）算法是一种动态的比特率控制策略，算法部署在 DASH 客户端播放器，根据当前的网络状况、客户端缓存状态和客户端本地硬件的处理能力等因素，动态地选择每个视频切片的比特率，目的是最大化用户的观看体验质量。

ABR 算法作为 DASH 技术的重要组成部分，直接影响着用户的观看体验。目前的 ABR 比特率算法主要分为四类：基于缓存的方法、基于带宽的方法、基于带宽和缓存因素的混合方法以及基于学习的方法。具体研究现状在本文 1.2 节

中进行了描述。

2.4 强化学习理论

强化学习就是学习“怎样做（如何把当前的环境状态映射成行为）才能使整个系统获得的累积奖赏（reward）最大化”[45]。强化学习能够在不知道环境模型和长期回报的情况下，智能体（Agent）通过与环境进行交互学习最优策略。其显著特征是 “试错”和延迟奖励。“试错”体现在智能体不断尝试动作，并与环境进行交互，根据环境的反馈指导后续动作[46]。延迟奖励体现在当智能体做出动作时不仅会获得即时奖励，也会影响下一个环境，从而影响后续奖励。

智能体与环境的具体交互方式如图 2-2 所示。智能体在 t 时刻根据当前的环境状态S_t，和上一时刻的奖励r_{t-1}选择当前时刻的动作a_t。执行动作a_t，并将其作用到环境中，环境会反馈一个该动作的即时奖励r_t，此时环境状态变为s_{t+1}。智能体在 t+1 时刻再次根据当前环境状态S_{t+1}和r_t继续悬着动作并执行，这个过程被重复执行。

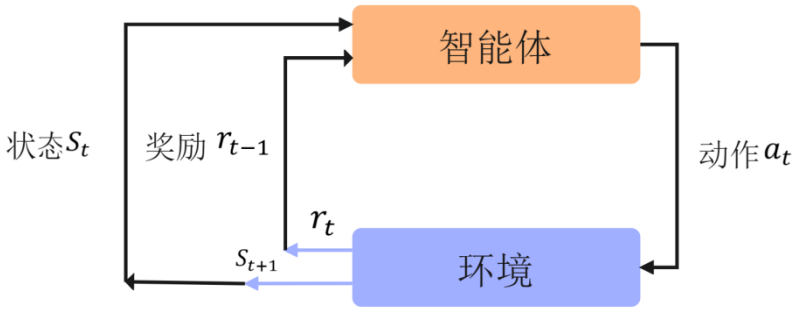


图 2-2 智能体与环境交互图

Figure 2-2 Interaction between the agent and the environment

通常情况下强化学习问题被建模为一个马尔可夫决策过程(Markov decision process，MDP)[47]。MDP 可以被表示为一个五元组$\langle S, A, P, R, \gamma \rangle$，其中,$S$表示智能体所有环境状态 s 的集合,$s_t$表示 t 时刻智能体的环境状态。A 表示智能体所能选择的动作a的集合，a_t表示智能体 t 时刻选择的动作。 P表示执行动作环境状态发生转移的概率，即状态转移概率。 R表示环境奖励r的集合。

r_t表示智能体在 t 时刻执行动作a_t环境给的奖励。$\gamma \in [0,1)$表示折扣系数，用来平衡当前和未来的奖励权重[48]。在 MDP 中,状态价值函数$V(s)$表示智能体

在状态 s 潜在的价值，它的值等于回报的期望。具体表示形式如式(2-1)。

$$V(s) = E[G_t|S_t = s] \tag{2-1}$$

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k} \tag{2-2}$$

其中G_t表示回报，表示从任务开始到结束，所有折扣奖励之和。G_t的表达式如式(2-2)所示。

由状态价值函数的表达式可以得到其递推公式，如式(2-3)所示。

$$V(s) = E[r_t + \gamma V(s_{t+1})|S_t = s] \tag{2-3}$$

进一步得到状态价值函数的贝尔曼方程：

$$V(s) = r(s) + \gamma \sum_{s'} p(s'|s)V(s') \tag{2-4}$$

动作价值函数(action-value function) $Q^{\pi}(s_t, a_t)$,表示智能体在状态s_t下执行动作a_t得到的期望累积奖励，它表示智能体在状态s_t下执行动作a_t的潜在价值。具体显现形式如式（2-5）

$$\begin{aligned} Q^{\pi}(s_t, a_t) &= E[r_t + \lambda r_{t+1} + \lambda^2 r_{t+2} + \cdots \mid s_t, a_t] \\ &= E[r_t + Q(s_{t+1}, a_{t+1}) \mid s_t, a_t] \end{aligned} \tag{2-5}$$

在强化学习中通常根据状态价值函数衡量一个状态的好坏，根据动作价值函数衡量一个动作的好坏，状态价值函数和动作价值函数共同帮助智能体决策。

在现实世界的众多场景中，多智能体系统的存在是普遍的。这些场景包括但不限于交通管理、智能电网、自动化仓库和机器人足球等。在这些复杂的环境中，单一智能体强化学习算法往往难以有效应对多智能体之间的交互和协作问题。因此，需要将其扩展至多智能体环境中[49]。

2.5 多智能体强化学习

多智能体强化学习（MARL）是强化学习（RL）领域的一个重要分支[50]，近年来因其在处理复杂交互和现实世界问题方面的潜力而受到广泛关注。与传统的单智能体强化学习主要关注单一智能体在环境中的学习和决策不同，多智能体强化学习专注于多个智能体在同一环境下的相互作用和协作。这种相互作用的复杂性，以及与现实世界任务的紧密贴合，使得 MARL 成为了研究的热点。强化学习的核心在于通过最大化从环境中获得的累计奖励来学习完成任务的最优策略。随着深度学习技术的发展，深度强化学习（DRL）算法在多种领域，

如棋类游戏、实时战略游戏、非完美信息游戏和自动驾驶等，展现出了卓越的性能[51]。然而，尽管单智能体强化学习在实际应用中取得了显著成就，多智能体场景下的交互和协作问题仍然是一个挑战。

多智能体强化学习尝试解决的是如何在多智能体共存的环境中，使每个智能体都能学习到最优或次优的策略，以实现个体目标和整体目标的最大化。这包括了智能体如何在没有完全环境信息的情况下做出决策、如何在智能体间分配任务和资源、以及如何在竞争和协作的情境中平衡个体利益和集体利益等问题。由于这些问题的复杂性，MARL 不仅在理论研究中占有重要地位，同时也在工业制造、机器人控制等实际应用领域展现出巨大的应用潜力。随着技术的进步和研究的深入，多智能体强化学习有望在更多领域中发挥关键作用，推动智能系统的发展[52]。

2.5.1 局部马尔科夫决策过程

局部马尔可夫决策过程（Partially Observable Markov Decision Process, POMDP）是一种用于描述在不完全信息环境下的决策问题的框架。与完全可观测的马尔可夫决策过程（MDP）不同，POMDP 考虑了决策者无法完全观察到环境状态的情况。在 POMDP 中，智能体只能接收到环境的部分观测，这些观测信息可能无法完全反映环境的真实状态。因此，智能体需要基于有限的观测信息来推断环境的状态，并在此基础上做出决策。POMDP 模型通过引入观测函数和信念状态的概念，允许智能体在有限信息的基础上进行学习和决策。信念状态代表了基于历史观测所形成的对环境状态的估计，智能体通过更新信念状态来适应环境的变化，从而在不确定性较高的环境中做出更加合理的决策。这种模型在处理现实世界中信息不完全的决策问题时具有重要意义，尤其适用于复杂的多智能体系统，其中每个智能体都可能只能获取局部信息，每个智能体都拥有自己的奖励函数[53]。其与环境的交互过程如图 2-3 所示:

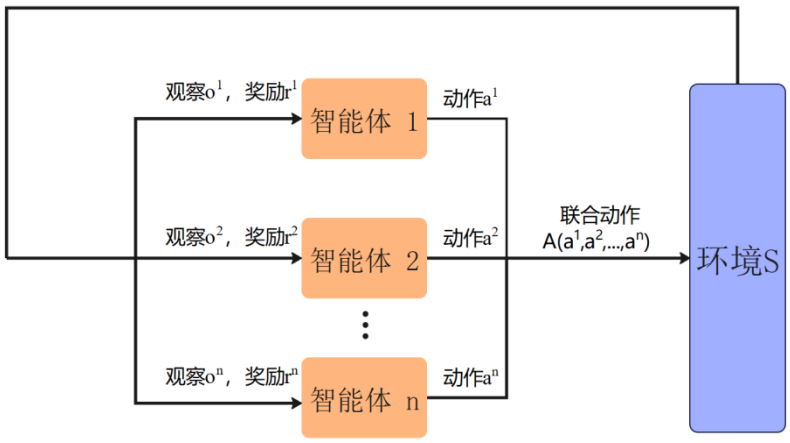


图 2-3 多智能体与环境交互

Figure 2-3 multi-agent interaction with the environment

POMDP 可以通过元组$< S, A, T, R, O, \Omega, \gamma >$表示，其中 S表示环境状态，它包括所有智能体的本地观测以及部分全局信息。$A: a^1, \ldots a^n$表示所有智能体的联合动作。T表示状态转移概率分布函数，$T(s, a, s') = Pr(s_t = s' | s_{t-1} = s, a_{t-1} = a)$表示状态 s 下执行动作$a$，状态转变为$s'$的概率。$R: r^1, \ldots, r^n$表示为各个智能体的奖励函数。$O: o^1, \ldots, o^n$表示所有智能体的本地观测。Ω表示观测概率分布函数，$\Omega(a, s', o) = Pr(o_t = o | a_{t-1} = a, s_t = s)$表示在 t-1 时刻执行动作 a 到达状态 s 观察到观测o的概率。γ为折扣系数，表示未来奖励相对于即时奖励的重要性。

2.5.2 多智能体强化学习框架

常见的多智能体强化学习方法可以分为集中式方法(Centralized Method）、分布式方法（Decentralized Method）和集中式训练分布式执行方法(Centralized Training with Decentralized Execution，CTDE)[54]。

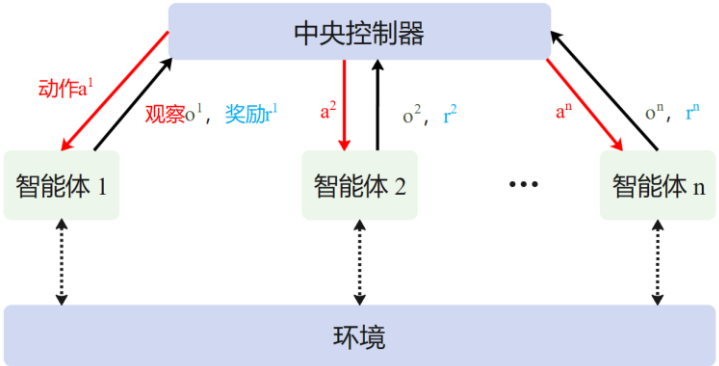


图 2-4 集中式方法

Figure 5-4 centralized method

集中式方法: 如图 2-4 所示，在集中式方法中所有智能体的决策过程集中到一个中央控制器，由此控制器处理所有的信息并指导每个智能体的行动。这种方法的主要优势在于能够全面考虑所有智能体的状态和动作，从而优化整体的决策和协作效率。通过集中处理和分析，可以更容易地实现复杂策略的协调和优化，尤其是在需要紧密协作的任务中。然而，集中式方法的缺点包括可能的计算瓶颈、难以扩展到大规模系统，以及对中央控制器的高依赖性，这可能影响系统的鲁棒性和灵活性。

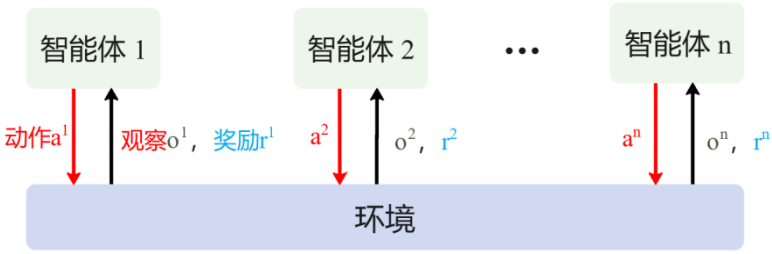


图 2-5 分布式方法

Figure 2-5 distributed method

分布式方法: 如图 2-5 所示，在分布式方法中每个智能体独立地根据自己的观测进行学习和决策，而不依赖于中央控制器或其他智能体的信息。这种方法的核心优势在于其高度的可扩展性和鲁棒性，因为智能体不需要全局通信，从而能够适应于动态变化的环境和大规模系统。分布式方法通过本地信息处理和决策，使得系统能够更灵活地响应环境变化，同时降低了通信成本和计算复杂度。然而，这种方法可能面临协调和合作策略学习的挑战，因为缺乏全局视角可能导致局部最优而非全局最优的解决方案。

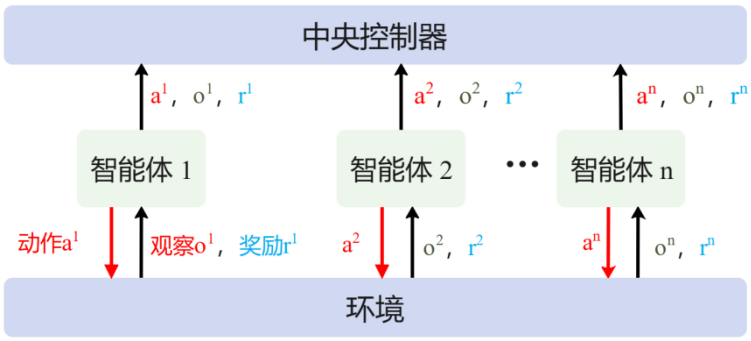


图 2-6 集中式训练分布式控制方法

Figure 2-6 centralized training of distributed control method

集中训练分布执行方法: 如图 2-6 所示，该方法结合了集中式方法和分布式

方法。核心思想是在训练阶段采用集中式，在执行阶段采用分散式。首先，在集中式训练阶段，所有智能体的本地观测 o 和动作 a 的信息被集合到一个中央控制器中，通过所有智能体的信息和部分全局信息(单个智能体无法获取的信息)对全局策略进行训练和优化。在分布式执行阶段，每个智能体仅需要根据训练好的策略和本地观察就可以执行动作，此时智能体仅能访问自己的本地信息。此方法中智能体在执行阶段无需全局通信和协调便能做出决策，因此具有很好的扩展性和稳定性。

集中训练分布式执行方法进一步可分为两类：基于 Actor-Critic 的方法和基于值函数分解（Value function decomposition）的方法[55]。

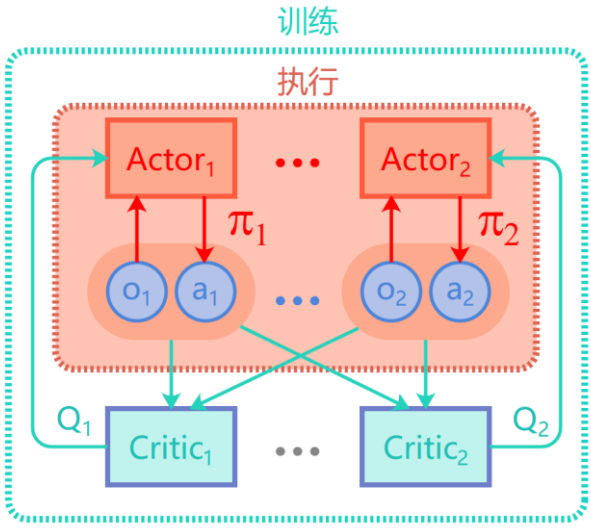


图 2-7 多智能体 Actor-Critic 方法

Figure 2-7 Multi-agent Actor-Critic method

基于 Actor-Critic 的方法：该类方法的基本思想是将强化学习中的 Actor-Critic 方法与多智能体强化学习相结合。如图 2-7 所示，在这种方法中，每一个智能体都有一个 Actor 网络，它用于更新智能体的策略网络π，智能体根据策略π选择动作。Critic 网络是评估智能体动作的价值网络，用于计算智能体在局部观察 o 下执行动作 a 的状态-动作价值(Q 值)。

Actor-Critic 方法中代表性的算法有 COMA[56]、MADDPG[57]等算法。MADDPG 算法允许每个智能体在训练时访问其他智能体的策略和状态信息，而在执行时只依赖于自身的观测和策略。这种设计使得MADDPG能够有效处理多智能体之间的协作和竞争问题。MADDPG 算法对每个智能体都使用两个网络：

一个 Actor 网络用于确定动作，一个 Critic 网络用于评估当前策略的价值。不同于单智能体的 DDPG，MADDPG 的 Critic 网络在评估某个智能体的动作价值时，会考虑到其他智能体的策略和动作，从而更好地理解多智能体环境中的动态和相互作用。这种机制使得MADDPG在处理复杂的多智能体交互问题时，能够学习到更加高效和协调的策略。MADDPG 算法的优势在于其能够在连续动作空间中处理多智能体之间的复杂交互，适用于需要精细动作控制的场景。此外，通过集中训练的方式，MADDPG 能够有效利用环境中的全局信息来指导每个智能体的学习，进而提高整体系统的性能。然而，MADDPG 算法复杂度高并且无法对智能体进行合适的信用分配，部分智能体之间性能差距较大。COMA 算法通过引入反事实基准（counterfactual baseline），对每个智能体的动作价值进行评估，从而准确地分配奖励给各个智能体。COMA 利用集中式训练，允许在训练过程中访问全局信息，而在执行阶段，每个智能体仅使用其局部观测进行决策。这种设计不仅提高了协作效率，还优化了学习过程，使得算法在多智能体协作任务中表现出色。

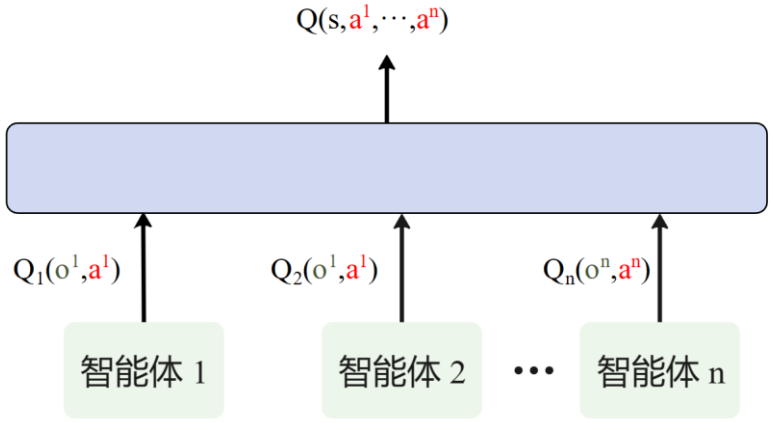


图 2-8 多智能体值函数分解方法

Figure 2-8 Multi-agent value function decomposition method

基于值函数分解的方法：这类方法的核心思想是将多智能体环境中的全局价值函数（也就是所有智能体共同行动产生的总体效用）分解为各个智能体的个体价值函数之和或其他组合形式[58]。如 2-8 图所示，在值函数分解方法中，每个智能体都有自己的局部动作值函数（局部 Q 值），表示该智能体在局部观测下采取动作的价值。联合动作值函数（全局 Q 值）由所有智能体的局部值函数共同组成，表示所有智能体在全局状态下的联合动作的价值。每个智能体的

局部值函数都是独立学习，且智能体的策略根据值函数而更新。基于值函数分解的方法中具有代表性的算法有 VDN[59]、QMIX[60]等算法。VDN 算法中简单的将全局 Q 值认为是所有智能体局部 Q 值的和，但这种方法可能导致在寻求全局最优解时只能达到局部最优。QMIX 算法使用一个单独的混合网络来整合各个智能体的局部 Q 值（代表每个智能体的行为价值），生成全局 Q 值（代表整个智能体群体的行为价值）。这种方法的关键在于，它约束了全局 Q 值是各个智能体局部 Q 值的单调递增函数，确保了在优化全局 Q 值时，任何智能体局部 Q 值的增加都不会导致全局 Q 值的减少。这样的设计促进了智能体间的有效协作，同时保持了策略的分布式执行能力，允许每个智能体根据局部观测独立作出决策。但 QMIX 算法的单调性约束可能不足以应对环境的快速变化，并且算法复杂度较高。

2.6 本章小结

本章首先介绍了流媒体传输技术，其中包括传统的流媒体传输技术和基于 HTTP 的自适应流技术，描述了传统流媒体传输技术的缺点和基于 HTTP 的自适应流媒体技术的发展过程。接着介绍了 MPEG-DASH 技术的主要内容，其中包括 MPEG-DASH 技术的框架，流媒体服务器和视频服务器，MPD 文件、视频切片和比特率控制算法以及 ABR 算法。最后介绍了强化学习理论和多智能体强化学习。在强化学习理论中介绍了强化学习概念、MDP 模型 以及值函数的计算。在多智能体强化学习中介绍了三种多智能体强化学习框架，并描述了各自的优缺点，最后详细介绍了基于集中式训练分布式执行的多智能体强化学习框架的多智能体深度强化学习方法，主要包括基于 Actor-Critic 的方法和基于值函数分解的方法。

第 3 章 吞吐量预测器 TPMI

本章针对现有的吞吐量预测器在无线环境下无法进行准确的吞吐量预测问题，设计并实现了一种基于 Informer[61]的吞吐量预测器 TPMI。该预测器的目标是能够实现对无线环境下吞吐量进行准确预测，以提升现有 ABR 算法在无线环境下的性能，从而提高用户在无线环境下的 QoE。本章首先进行了问题描述，其次介绍了 Informer 模型，接着进行了预测器设计，其中包括特征选择和基于 Informer 的吞吐量预测器，然后将 TPMI 集成到现有 ABR 算法中，以提升现有 ABR 算法的性能 ABR 算法中的性能进行了实验验证，并对实验结果进行分析总结。最后在仿真平台上，对所设计预测器的准确性和集成到现有

3.1 问题概述

在现有的 ABR 算法中，基于带宽和混合方法的 ABR 算法依赖于吞吐量预测。吞吐量的高估可能导致 ABR 算法选择更高比特率，从而引起视频卡顿。吞吐量的低估会导致 ABR 算法采取更保守的方法，选择较低比特率的视频切片，这不仅导致视频质量较低，还浪费了带宽资源。这两种情况都严重影响了用户的 QoE。基于缓存和学习的方法在实际部署中，仍然需要吞吐量信息的辅助。例如，BOLA 算法主要根据缓存信息选择比特率。然而，在实际部署时，采用了 DYNAMIC 方法，主要是因为 BOLA 算法在播放启动阶段性能表现不佳。DYNAMIC 在启动阶段利用吞吐量预测器信息进行比特率选择，并在达到一定缓冲水平后切换到 BOLA 算法。类似地，基于学习的 Pensieve 方法将过去 K 个视频切片的下载信息输入到深度强化学习模型中，经过 1D-CNN 处理后间接预测吞吐量。因此，吞吐量预测对 ABR 算法至关重要，准确的吞吐量预测可以显著提高 ABR 算法的性能。

然而无线网络带宽波动具有随机性、时变性，在无线环境中实现准确的吞吐量预测仍然是一个具有挑战性的任务。如图 3-1 所示，本章在无线网络环境中测试了 MPC 算法的比特率选择。实验结果显示 MPC 的比特率选择与实际可用带宽存在显著差异，与文献[62]中的描述一致。本章的研究发现这种现象不仅存在于 MPC 算法中，而且在大多数现有的 ABR 算法中也存在。这是由于视频

内容编码的限制，但比特率选择仍然依赖于吞吐量。例如，在 RobustMPC 中，展示了不准确吞吐量预测对用户 QoE 的影响。具体而言，如果一个切片的实际吞吐量突然下降，RobustMPC 中下一个切片的预测吞吐量也会减少；然而，当切片的实际吞吐量突然上升时，预测的吞吐量无法及时响应。尽管实际吞吐量足以支持最高比特率，RobustMPC 仍然选择较低的比特率，导致视频质量下降和波动，严重影响了用户 QoE。过去的研究通常将吞吐量变化归因于网络带宽的波动，并认为吞吐量难以预测。这是由于将应用吞吐量与可用带宽混淆，从而导致对吞吐量变化原因的误解。

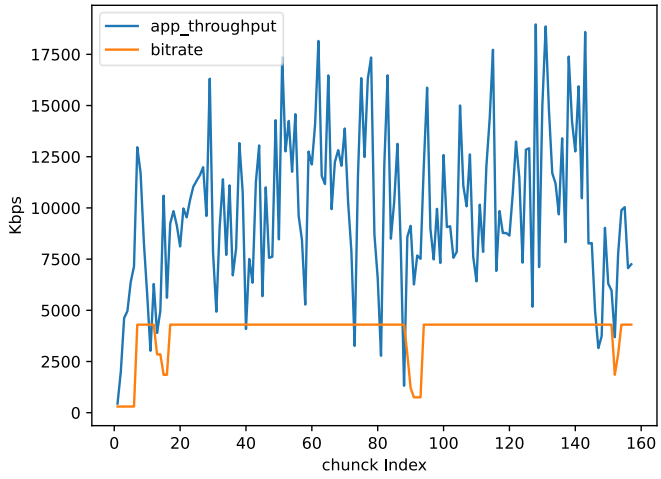


图 3-1 视频切片比特率和应用吞吐量

Figure 3-1 video Slice bit rate and application throughput

虽然吞吐量的预测已经得到广泛研究，但大多数现有的预测方法[63][64]忽略了影响吞吐量的各种因素，如吞吐量与历史吞吐量、切片大小、网络连接类型、信号强度和播放器状态的相关性。尽管一些研究专注于这些影响因素，它们通过计算吞吐量与影响因素之间的互信息来表示相关性。这些研究表明，影响吞吐量预测的因素不仅受网络条件的影响，还受切片大小、网络连接类型、信号强度和播放器状态的影响。尽管这些影响因素已被理解，但如何利用这些信息进行准确预测仍然是一个重大挑战。注意力机制可以更好地表示这些影响因素与吞吐量之间的依赖关系，因此本章考虑如何利用注意机制进行更准确的预测。

3.2 Informer 模型

Informer 是基于注意力机制的深度学习模型，它可以看作是 Transformer[65] 模型的进阶版，整体上由 Encoder 和 Decoder 两部分组成。其中编码器用于捕获原始输入序列之间的长期依赖关系，解码器可进一步实现序列预测。与 Transformer 相比 Informer 模型的改进主要体现在一下几点：

Transformer 计算 self-attention 时需要计算一个时刻与其他所有时刻的相似度，从而导致时间复杂度为 O(N^2)，针对这一点 Informer 提出了 prob-sparse self-attention 机制，使得时间复杂度降为 0($N\log N$)。

Transformer 输入长时间序列需要堆叠多个多头注意力机制，从而导致空间复杂度过高，Informer 针对这一点提出了 self-attention Distilling 机制，缩短每一层的输入序列长度，有效减小了计算和存储量。

Transformer 使用自回归式的预测，及采用逐步预测，这使得在长序列时间序列预测时预测速度下降，Informer 提出了生成式的 Decoder 机制，将预测时间复杂度由 0(N)降到了 0(1)。

Informer 模型的这些改进使得它在 LSTF(long sequence time-series forecasting) 问题上取得了不错的效果。吞吐量预测主要体现在吞吐量在一定时间范围内的变化，因此吞吐量预测可以看作是一个 LSTF 问题。本章基于 Informer 构建吞吐量预测器，它能够更好的捕捉输入特征信息之间的关系，实验结果表明该预测器具有较好的预测性能。

3.3 预测器设计

3.3.1 特征选择

本章最初通过研究确定了影响视频切片吞吐量预测的潜在因素。这些因素包括服务器的下行带宽、历史吞吐量、连接状态、目标切片大小和播放器状态。由于服务器的下行带宽对客户端来说很难获取，因此本章的研究不将其视为预测的输入特征。现有的研究主要关注整体网络特征来表示连接状态，如有关 ISP、AS 和 CDN 的信息[66][67]。然而，客户端无法直接获取这些信息。因此，本章选择用连接类型和信号强度来表示用户的网络情况。根据 WiFi RSSI 和 4G RSRP 将信号强度分为强、中和弱。此外，将播放器状态分为稳定和缓冲，稳定表示

视频正常播放，缓冲表示视频卡顿。本章还评估了吞吐量与这些因素之间的互信息，确定了影响切片吞吐量预测的主要因素，包括历史吞吐量、目标切片大小、连接类型、信号强度和播放器状态。本章选择了这些主要影响因素作为预测器的特征输入。

3.3.2 基于 Informer 的吞吐量预测器

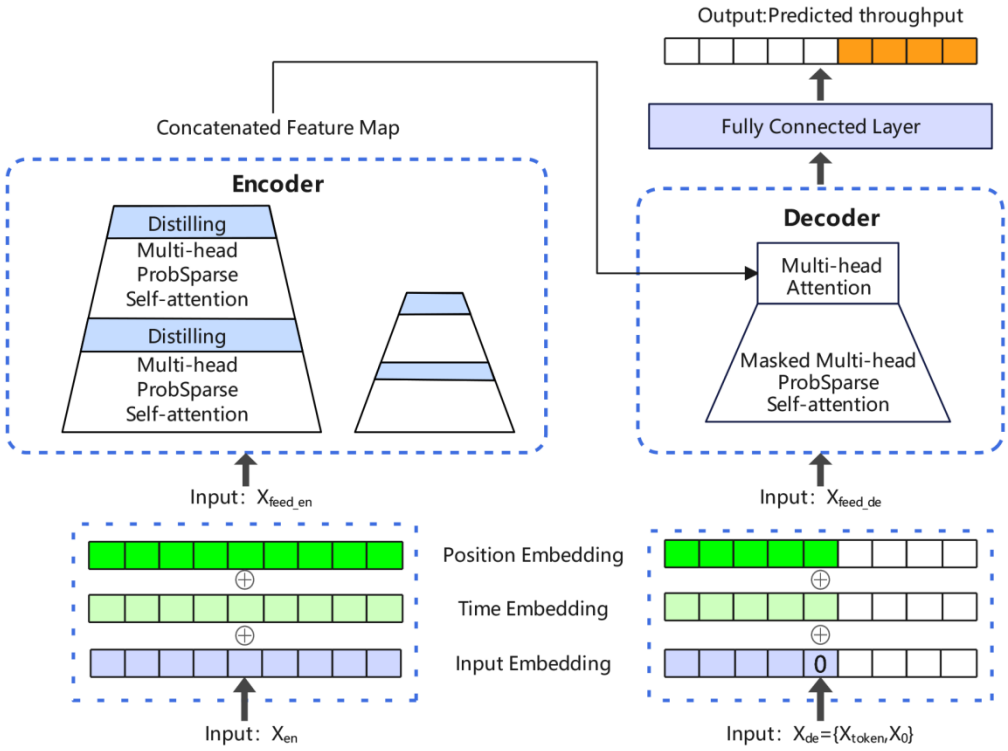


图 3-2 基于 Informer 的吞吐量预测器

Figure 3-2 Informer-based throughput predictor

基于 Informer 的吞吐量预测模型如图 3-2 所示，由四个部分组成：特征处理模块、编码器、解码器和全连接层。为了更好地探索数据集的结构并获得更多信息，预测模型首先处理原始数据集以进行特征处理。这旨在通过以下特征编码方法来减少计算时间并提高模型的预测准确性。

$$X_{en[i]}^{t} = (connection\_type, signal\_strength, chunk\_size, throughput, player\_stat) \quad (3-1)$$

$$u_i = Embedding\left(X_{en[i]}^{t}\right) \quad (3-2)$$

$$X_{feed\_en[i]}^{t} = \alpha u_i^t + PE_{(L_x\times(t-1)+i,)} + \sum_{p}\left[SE_{(L_x\times(t-1)+i)}\right]_p \quad (3-3)$$

在编码器中，利用了 prob-sparse self-attention 和 self-attention Distilling 机制。

原始特征向量包括连接类型、信号强度、切片大小、吞吐量和播放器状态。$X^t_{feed_en[i]}$表示通过特征处理后特征向量。PE 代表位置嵌入，其具体形式如方程(3-4)所示。SE 代表时间嵌入。

$$\begin{aligned} PE_{(pos,2j)} &= sin\left(\frac{pos}{(2L_x)^{\frac{2j}{d_{model}}}}\right) \\ PE_{(pos,2j+1)} &= cos\left(\frac{pos}{(2L_x)^{\frac{2j}{d_{model}}}}\right) \end{aligned} \quad (3-4)$$

接着将处理后的序列长度为 Lx 的特征序列

$$X^t_{feed\_en} = \{X^t_{feed\_en[1]}, \cdots, X^t_{feed\_en[L_x]} | X^t_{feed\_en[i]} \in R^{dx}\} \quad (3-5)$$

输入到 Informer 的编码架构中，从而捕捉特征序列输入与输出之间的长期相关性。原始输入特征序列$X^t_{de} = Concat(X^t_{token}, X^t_{on})$在解码器中的特征处理形式与编码器中的相同。模型 Decoder 的输入数据为：

$$X^t_{feed\_de} = Concat\left(X^t_{feed\_token}, X^t_{feed\_on}\right) \in R^{(L_{token}+L_y)\times d_{model}} \quad (3-6)$$

$X^t_{feed_token}$表示开始 Token，它是靠近预测目标的输入序列中动态采样的部分序列经过 embedding 后的数据。$X^t_{feed_on}$表要预测的吞吐量，初始化为 0。Decoder 的输出通过全连接层获得预测结果，表示为:

$$Y_{out} = \{Y_1, \cdots, Y_{Ly} | Y_i \in R^{dy}\} \quad (3-7)$$

这里，Y_i表示 t 时的预测吞吐量。本文已经在收集的数据集上训练了模型，并测试了其性能。

3.4 基于 TPMI 的 ABR 算法

本章将设计并实现的吞吐量预测器，命名为 TPMI。TPMI 可以集成到任何 ABR 算法中，该算法使用吞吐量预测作为移动自适应流中的吞吐量预测器。本章使用经典的 ABR 算法 MPC 作为例子来展示 TPMI 是如何应用于现有方案的。

作为自适应流媒体中视频切片的吞吐量预测器，TPMI 可以集成到任何使用吞吐量预测的 ABR 算法中。为了评估 TPMI 在集成到现有算法时的性能，本章将其整成到 MPC 算法中，这是一种基于吞吐量预测的 ABR 算法。MPC 根据缓

冲区和通过 HM 预测器（过去 5 个样本的调和均值）预测的吞吐量预测模拟未来片段的 QoE，然后选择比特率。MPC 的理论性能提升与其吞吐量预测的准确性成正比。本章将 TPMI 集成到 MPC 算法中，算法伪代码如表 3-1 所示，通过将 MPC 中的 HM 预测器替换为 TPMI 来预测未来每个切片的吞吐量，其中R_k的计算方式如式(3-8)和(3-9)所示，表 3-2 列出了各个变量的表示。

$$R_k = arg\max_{r,1\le j\le m} \sum_{l=k}^{k+n-1} Q\hat{o}E\left(R_{l|r_j}\right) \tag{3-8}$$

$$\begin{aligned} Q\hat{o}E(R_{l|r_j}) = R_{l|r_j} - max\left(\mu(\frac{d_l\left(R_{l|r_j}\right)}{\hat{T}_{l|r_j}} - B_l),0\right) \\ -\lambda\left|R_{l|r_j} - R_{l-1}\right|, k \le l \le k+n-1 \end{aligned} \tag{3-9}$$

表 3-1 基于 TPMI 的 MPC 算法伪代码

Table 3-1 Pseudocode of the MPC algorithm based on TPMI

Algorithm 基于 TPMI 的 MPC 算法

Initialize
for k = 1 to K **do**
    if 播放器处于缓冲状态 **then**
        $\hat{T}_{[t_k,t_{k+N}]}$ = TPMI($T_{[t_1,t_k]}$,连接类型，信号强度，块大小，播放状态)
        $[R_k, T_s] = f_{mpc}^{st}(R_{k-1}, B_k, \hat{T}_{[t_k,t_{k+N}]})$ //比特率选择和计算播放等待时间
        等待T_s秒开始播放
    else if 播放器处于稳定状态 **then**
        $\hat{T}_{[t_k,t_{k+N}]}$= TPMI($T_{[t_1,t_k]}$,连接类型，信号强度，块大小，播放状态)
        $R_k = f_{mpc}(R_{k-1}, B_k, \hat{T}_{[t_k,t_{k+N}]})$ //比特率选择
    end if
    以比特率R_k下载第 K 个视频切片
end for //下载结束

表 3-2 变量及含义表

3-2 Variables and their meanings

变量	含义
R_k	第 k 切片的比特率
m	每个视频切片的比特率数量
$R_{k\|r_j}$	第 k 个视频切片的比特率r_j
$\hat{T}_{l\|r_j}$	比特率r_j下吞吐量的预测值
$d_k(R_{k\|r_j})$	比特率为r_j第 k 个视频切片的大小

3.5 实验与分析

3.5.1 实验设置

实验环境：PC 机配置为 Nvidia 1080ti 11G、32G 内存、64 位 Windows11 操作系统作为实验平台，Python3.8、Pytorch1.8.0 等开发工具。

视频会话数据集：本章使用 Lumos[68]中的数据集，并在此基础上进行数据预处理，最终得到包含 590 个的视频会话（392 个运行 ABR 算法的会话）和恒定比特率级别（198 个会话）。其中 80%用于 TPMI 训练，20%用于测试。每个会话通常有 5 分钟或 5.5 分钟的播放持续时间。网络连接类型包括"wifi_2.4GHz", "wifi_5GHz", "4g"，信号强度划分为["强", "中", "弱"]三个等级。实验视频名称为 Big Buck Bunny 和 Elephant Dream，每个视频切片持续时间为 2s。视频文件编码为 4 个质量等级，比特率分别为 300、1200、2850、4300kbps。播放状态分为缓冲状态和稳定状态。

3.5.2 评价指标

TPMI 预测性能评估：本章选取了三种常用方法作为比较基准：多重线性回归（MLR）、过去 5 个样本的谐波平均值（HM）和长短期记忆网络（LSTM）。预测准确性的评价指标为：利用预测误差和均方误差（MSE）来评估吞吐量预测的准确性。

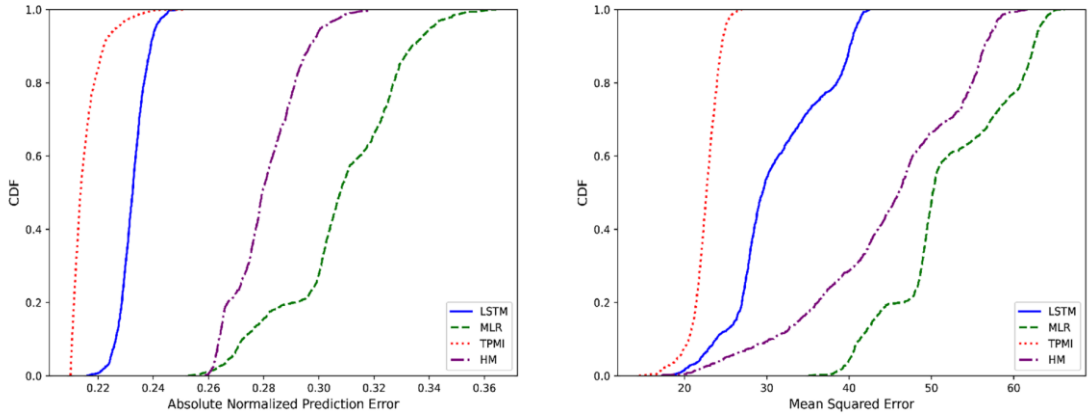
MPC+TPMI 的性能评估：为了评估 MPC+TPMI 的性能。本章将其性能与 MPC、BBA 和 Pensieve 进行了评估。具体评价指标如下所示：

(1)QoE 指标：为评估三种自适应比特率（ABR）算法的用户体验质量（QoE），本章采用了 MPC 中定义的度量标准，具体如下：

$$QoE = \sum_{k=1}^{N} R_k - \mu \sum_{k=1}^{N} max\left(\left(\frac{d_k(R_k)}{T_k} - B_k\right), 0\right) - \lambda \sum_{k=1}^{N-1} |R_{k+1} - R_k| \quad (3-10)$$

其中 N 是会话中的视频切片数量，R_k是客户端选择的第 k 个视频切片的比特率，第二项表示传送第 k 个片段的重新缓冲时间，最后一项表示片段之间比特率切换的平滑程度。在这个方程中，μ 和 λ 是非负的加权参数

(2)平均比特率和重新缓冲指标：为评估三种自适应比特率（ABR）算法，本章在评估 QoE 的基础上同时进行了平均比特率和重新缓冲比较。



(a) (b)

图 3-3 两种指标的预测性能。(a) 预测误差；(b) 均方误差

Figure 3-3 Prediction performance of the two methods. (a) absolute normalized pridiction error; (b)mean squared error

3.5.3 结果与分析

图 3-3 展示了这两种方法的预测准确性。实验结果表明，无论采用何种度量标准，TPMI 在吞吐量预测方面表现最佳。具体而言，相较于其他方法，TPMI 将误差降低了 16.8%至 38.7%。值得关注的是，在强网络条件下，TPMI 的平均预测误差仅约为 5.5%，较其他方法提高了 57.7%至 74.0%。即便在弱网络条件下，TPMI 也将准确性提升了 9.1%至 30%。

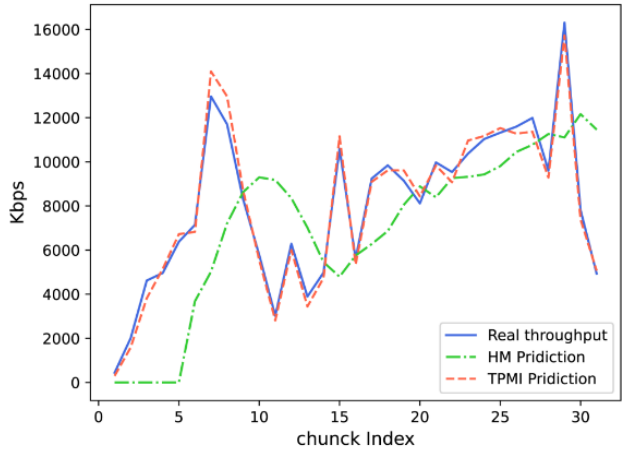


图 3-4 TPMI 对波动的快速反应

Figure 3-4TPMI's rapid response to fluctuations

如图 3-4 所示，在缺乏足够历史数据的情况下，例如会话启动阶段，HM 由于缺乏足够历史数据而无法进行吞吐量预测。而 TPMI 通过考虑网络条件和播

放器状态，在相同情境下将前五个片段的平均预测误差降至仅 12.8%。在这种情况下，TPMI 通过对波动的快速反应超过了 HM。即便有足够的历史吞吐量数据，当吞吐量从 1.2Mbps 突然下降至 0.32Mbps，并持续下降时，由于 HM 对异常值不敏感，它应该降低预测值，但实际上却连续几个片段增加了预测值，这可能会导致视频重新缓冲。准确的吞吐量预测器必须能够迅速捕捉到此类变化并做出及时响应。TPMI 通过对网络条件和播放器状态的感知，能够迅速响应这些变化，做出更准确的预测。以上结果表明，TPMI 能够有效地学习吞吐量与影响因素之间的依赖关系，并具有更好的预测性能。

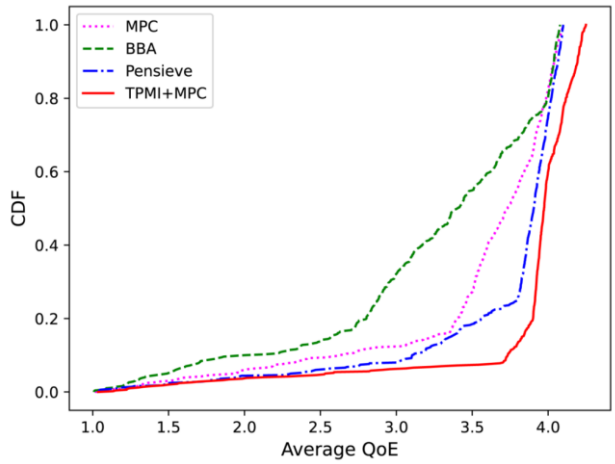


图 3-5 平均 QoE

Figure 3-5 Average QoE

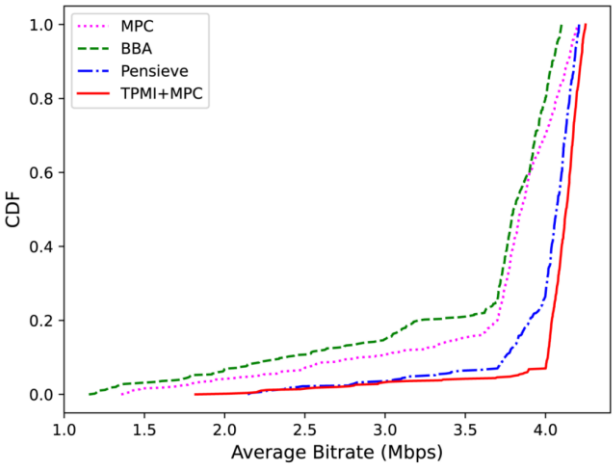


图 3-6 平均比特率

Figure 3-6 Average bitrate

图 3-5 显示了 MPC、MPC+TPMI、BBA 和 Pensieve 的 QoE 性能。总体而言，与 MPC 相比，TPMI 提高了平均 QoE 分别达到 9.67%，19.16%和 4.5%。图

3-6 和图 3-7 显示了，MPC+TPMI 在保持可接受的重新缓冲时间的同时总是选择较高的比特率。此外，与 MPC 相比，MPC+TPMI 平均比特率增加了 9.74%，同时将重新缓冲时间减少了 26.1%。这些结果更加证明了 TPMI 准确预测的重要性，准确的吞吐量预测可以有效提升 ABR 算法的性能,从而为用户带来更好的观看体验质量。

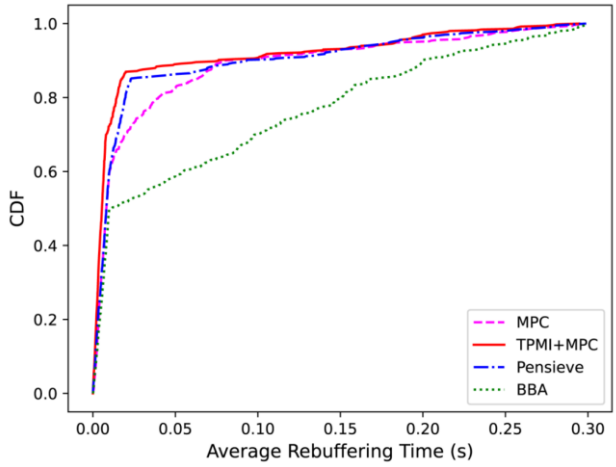


图 3-7 平均重新缓冲时间

Figure 3-7 Average rebuffering time

3.6 本章小结

本章首先描述了吞吐量预测器研究的目的和意义，并阐述了本章研究的动机。然后介绍了 Informer 模型并描述了本章选择 Informer 模型的原因。随后本章对影响吞吐量预测的因素进行了分析，针对现有预测器的不足，本章设计并实现了一种准确的吞吐量预测器，命名为 TPMI，可以集成到任何考虑吞吐量为因素的 ABR 算法中，以提高 ABR 算法的性能。该预测器基于 Informer 模型构建，考虑了影响吞吐量预测更广泛的因素作为输入特征，例如历史吞吐量、切片大小、网络的连接类型、信号强度和播放器状态等。本章评估了 TPMI 预测性能，并将其与基于 HM、MLR 和 LSTM 的预测方法进行了比较，预测性能明显提升。最后将 TPMI 集成到现有的 ABR 算法 MPC 中，并评估了 TPMI+MPC 的性能，与原始 MPC 算法相比，QoE 增加了 9.67%，与 Pensieve 和 BBA 相比平均 QoE 比提高了 4.5%到 19.16%。实验结果表明本章所设计的吞吐量预测器 TPMI 具有良好的预测性能，可以明显提高现有 ABR 算法的性能，实现为无线环境下流媒体用户提供更好的 QoE。

第 4 章 基于 QoE 感知的替换补偿算法

本文在第三章节中提出了一种基于 Informer 的吞吐量预测器，虽然能够实现在剧烈且频繁波动的无线网络环境中对吞吐量的准确预测，但不能改变实际的无线网络环境。本章的目标是提出一种优化算法实现对现有 ABR 算法的性能提升，旨在解决现有 ABR 算法在带宽剧烈且频繁波动的无线环境下，视频质量和平滑度方面低的问题，以便在无线环境中尽可能提高用户的 QoE。

本章首先分析了无线流媒体 QoE 的影响因素。其次进行了问题描述，主要体现在现有 ABR 算法在无线环境下的不足。随后阐述了本章算法的系统模型，然后进行了算法设计，并将所提算法融合到现有 ABR 算法中，以提升现有 ABR 算法性能。最后在仿真平台上，对所提算法进行实验验证并进行结果分析。

4.1 无线流媒体 QoE 影响因素

在移动流媒体不断发展的过程中，用户观看视频的 QoE（体验质量）是研究者关注的重点问题。文献[12]指出，视频的启动时间、视频的平滑度、视频的卡顿、视频的质量等都会影响用户的 QoE。视频的启动时间是指用户点击播放到能看到画面的时延；视频的平滑度是指不同比特率之间的切换次数和幅度；视频的卡顿是指视频重新缓冲的时间，在此阶段视频播放停止；视频的质量是指视频的比特率，比特率越高，质量越高。理论上，ABR 算法应该满足低启动时间和低卡顿频率，同时保持高视频质量和平滑度。

但是实际中，这些因素之间存在着制约关系。例如，为了匹配波动的带宽，频繁切换比特率虽然在一定程度上可以提高视频平均比特率，但同时会造成重新缓冲的风险。在启动阶段，一般为了便于快速启动会选择低比特率视频，虽然减少了启动时间，但会造成低清晰度的播放。同样，如果在启动阶段选择高比特率视频会增加用户的等待时长，降低用户对视频内容的兴趣。启动缓慢、卡顿、以低比特率播放、比特率波动幅度大、比特率变换频繁等事件，都会极大地降低用户的 QoE，从而导致用户观看完整视频的兴趣度会大大降低，很可能会放弃观看。因此，ABR 算法的设计需要综合考虑影响视频用户 QoE 的各个因素，尽量满足在提高平均比特率的同时尽量减少重新缓冲和比特率切换。

4.2 问题描述

无线网络带宽的变化存在随机性，带宽波动大，现在的自适应算法在低带宽时选择下载低比特率视频切片，带宽升高只影响未来比特率选择，缓冲区中仍存在低比特率视频切片，这极大影响了视频质量，从而降低了用户的 QoE。

具体而言，现有的 ABR 算法，将视频下载看作是一个顺序决策过程，只决定下一视频切片的比特率选择，已被下载的视频切片比特率不能够改变。当网络带宽剧烈频繁波动时，他们的性能表现不佳。例如当网络带宽由 6Mbps 突然下降至 1Mbps，且持续一段时间，此时现有的基于带宽的 ABR 算法，会选择下载低比特率视频，同样在面临这种情况时，现有的基于缓冲区的 ABR 算法，虽然不会骤然发生比特率下降，但实际当这种波动频繁时，在低带宽期间，缓冲区被严重消耗，缓冲区占比下降，此时同样会选择低比特率视频。即使网络带宽再次上升至 6Mbps 时，此前下载的低比特率视频不会被改变，缓冲区中仍是低比特率视频。为了模拟这种网络波动带来的影响，本章合成了一个网络带宽轨迹，网络带宽周期性变动，变化范围从 1Mbps 到 6Mbps。如图 4-1 所示，基于缓冲区的自适应算法 BOLA-E[69]和基于带宽的自适应算法 THROUGHPUT[23]在带宽剧烈频繁波动下，视频切片下载比特率同样波动严重，这极大的影响了视频的播放质量和播放平滑度。

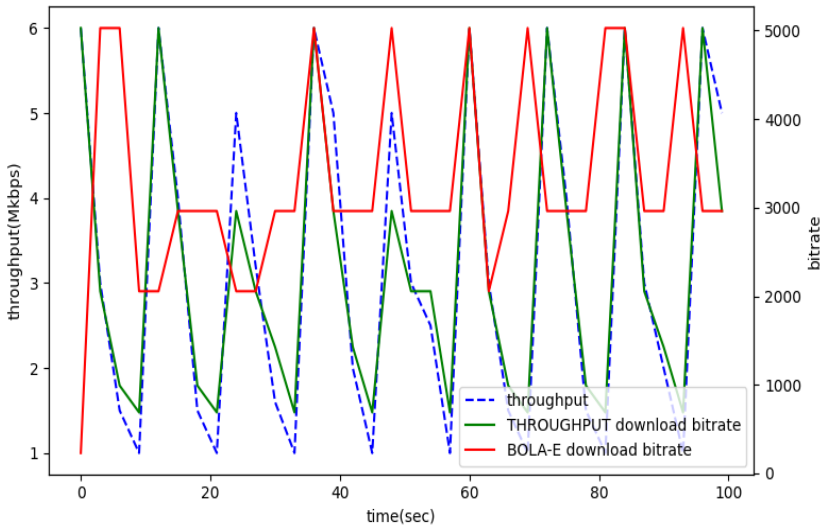


图 4-1 带宽波动对 BOLA-E 和 THROUGHPUT 比特率选择的影响

Figure 4-1 Influence of bandwidth fluctuations on the selection of BOLA-E and THROUGHPUT bitrate

4.3 系统模型

系统模型如图 4-2 所示。该系统模型基于 DASH 框架，主要由流媒体客户端和流媒体服务器端、CDN 组成。其中流媒体服务器端的视频切片和 MPD 文件与 DASH 中描述一致。流媒体客户端，主要由四个模块组成，分别为带宽预测模块 throughput predictor、缓冲区模块 playback buffer、视频播放模块 video play、比特率控制模块 ABR control。本章提出的算法部署在客户端的 ABR control 模块，包含两种下载决策，新视频切片下载决策 new segment download policy 和替换决策 replacement policy。首先客户端从视频服务器获取 MDP 文件，解析接收到的 MDP 文件，同时 ABR control 模块根据带宽估计模块估计的带宽和缓冲区模块提供的缓冲区占比进行决策，动态地选择最合适的比特率分片，然后客户端向 CDN 发送所需下载的视频切片以及比特率进行下载,下载的视频分片将存储在客户端的 playback buffer 中，再由 video play 模块进行视频切片的播放。

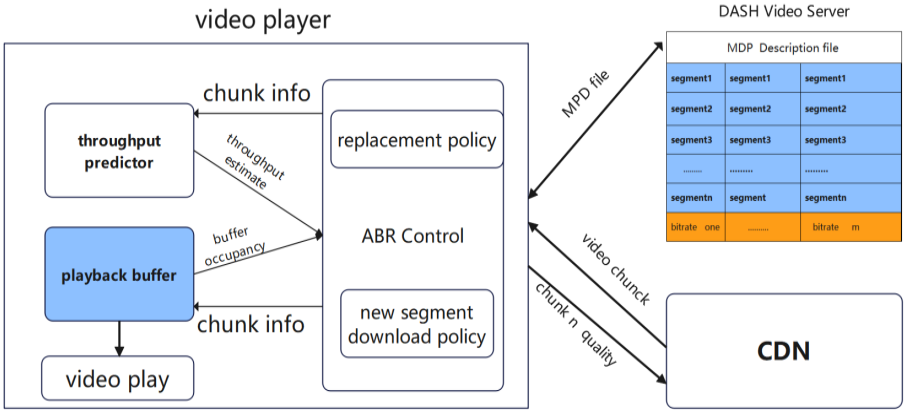


图 4-2 系统模型图

Figure 4-2 Diagram of the system model

4.4 算法设计

4.4.1 算法思想

与现有的 ABR 算法相比，本章的算法不同点在于能够对之前的比特率决策进行补偿。算法分为新视频切片下载和替换，新视频切片下载可以是基于带宽的自适应算法或者是基于缓存的自适应算法，替换是指下载更高比特率版本视频切片替换缓冲区中尚未播放的低比特率视频切片。算法的目的是通过替换提

高用户的 QoE。替换不会导致缓冲区的增加，这有可能导致卡顿，为了减少替换引起卡顿的可能性，算法将视频缓冲区分为三部分，缓冲区占比小于B_{low}，算法只执行新视频切片下载，缓冲区占比在B_{low}和B_{high}之间，算法根据当前的带宽，择机执行新视频切片下载和替换，同时为了尽可能多的替换，缓冲区占比大于B_{high}，算法根据当前的带宽只决定是否替换。同时为了避免资源浪费，替换必须在尚未播放前完成。

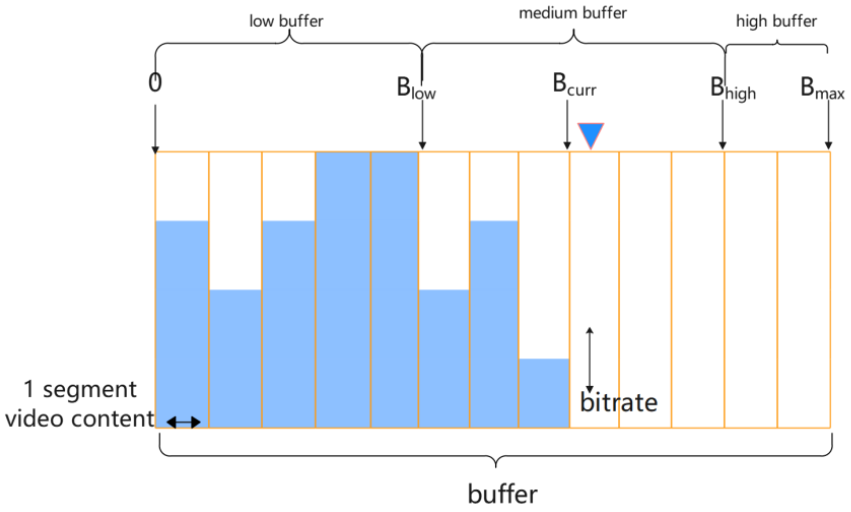


图 4-3 视频缓冲区阈值设置

Figure 4-3 Setting the threshold of the video buffer

4.4.2 缓冲区阈值

客户端缓冲区阈值设置,如图 4-3 所示，等间距的矩形垂直条表示一段 p 秒的视频,视频条的高度表示此视频切片的比特率。越高，视频的比特率越高，视频切片清晰度越高，视频切片数据量越大信息越多。同时将划分为三部分，分别为 low buffer 段、medium buffer 段、和 high buffer 段。B_{cuur}为当前缓冲区占比，B_{low}和B_{high}为算法缓冲区的两个阈值。low buffer 段B_{cuur}<B_{low}，medium buffer 段B_{low}<B_{cuur}<B_{high}，high buffer 段B_{high}<B_{cuur}<B_{max} (B_{max}为最大缓冲区占比,恒为 1)。

4.4.3 替换策略

首先算法计算缓冲区占比，然后，由 ABR control 模块根据缓冲区占比决定执行替换或者是新视频切片下载。当缓冲区占比处于 low buffer 段时，带宽预测模块对当前的带宽进行估计并作为可用带宽，算法启动新视频切片下载。当缓冲区占比处于 medium buffer 段时，判断此时带宽是否上升，若上升，算法启用替换，否则继续执行新视频切片下载。当缓冲区占比处于 high buffer 段时，同样判断此时带宽是否上升，若上升，选择执行替换，否则等待 Δ 秒在进行下一切片的下载决策。带宽上升定义为当前允许最大比特率b_{max}是否大于缓冲区中视频切片的平均比特率,大于则表明带宽上升。缓冲区视频切片的平均比特率表示为b_{ave}。

$$b_{ave} = \sum_{i=1}^{l} b_i \tag{4-1}$$

l表示缓冲区中视频切片数, b_i表示缓冲区中每个视频切片的比特率。本章根据替换效用函数来选择缓冲区中的视频切片进行替换，效用函数设计如下：

$$\max_k \left\{ \sum_{i=1}^{l} b_i + b'_k + \mu g \right\} \tag{4-2}$$

$$g = \sum_{i=1}^{k-1} f_{(i)} - \frac{S_{b'_k}}{T} \tag{4-3}$$

$$subject\ to\ \ g > 0; b_k < b_{ave}; b'_k > b_k \tag{4-4}$$

$$b'_k, b_k \in \{230,331,477,688,991,1427,2056,29625027,6000\} kbps \tag{4-5}$$

其中b_i表示缓冲区中的视频切片比特率，b'_k表示替换缓冲区中第k切片视频的比特率，b_k表示替换之前缓冲区中第k视频切片的比特率, $S_{b'_k}$表示以用来进行替换的视频切片大小，T表示此时的吞吐量, $f_{(i)}$表示缓冲区中第i个视频切片的播放时间,恒为 p 秒。g 表示替换完成时，缓冲区去中的原视频切片是否已经播放，g>0 表示尚未播放。若替换会引起卡顿,则算法根据缓冲区占比决定是否继续执行新视频切片下载。若B_{low}<B_{cuur}<B_{high}，则继续执行新视频切片下载，否则等待 t 秒。算法伪代码如表 4-1 所示：

表 4-1 算法伪代码

Table 4-1 Algorithm pseudocode

Algorithm 基于 QoE 感知的替换补偿算法

Data：$B_{curr}, B_{max}, B_{low}, B_{high}, K$表示视频切片索引，$Q_k$表示比特率等级
Initialization buffer Threshold parameters $B_{max}=1$，$B_{low}=0.6$，$B_{high}=0.9$
开始
  以最小的比特率下载第一个视频切片
  for i =1 to N (视频切片数量) do
    获取缓冲区占比B_{curr}
    if B_{curr} <B_{low} then
      根据 new segment download policy 获取 K 和Q_k //新视频切片下载
    else if B_{low}<B_{curr}<B_{high} then
      计算缓冲区视频切片平均比特率b_{ave}
      根据当前缓冲区占比和吞吐量获得允许下载的最高比特率b
      if b_{ave}< b then
        if 替换没有引起重新缓冲 then
          根据 replacement policy 获取 K 和Q_k //替换缓冲区中尚未播放的视频切片
        else
          根据 new segment download policy 获取 K 和Q_k
        end if
      end if
    else if B_{high}<B_{curr}<B_{max} then
      计算缓冲区视频切片平均比特率b_{ave}
      根据当前缓冲区占比和吞吐量获得允许下载的最高比特率b
      if b_{ave}< b then
        if 替换没有引起重新缓冲 then
          根据 replacement policy 获取 K 和Q_k
        else
          等待△T 秒 //缓冲区足够大，但替换仍然会引起卡顿
          根据 new segment download policy 获取 K 和Q_k
        end if
      end if
    end if
  以Q_k下载视频切片 K //执行下载
  更新缓冲区
  end for //下载结束
结束

4.5 实验与分析

在本节中，为了验证所提算法的有效性，将所提算法分别融合到基于缓冲区的 ABR 算法 BOLA-E 和基于带宽的 ABR 算法 THROUGHPUT 中进行实验评估。本节首先描述了实验环境和评价指标，然后给出了实验结果和实验分析。

4.5.1 实验环境

本章以 sabre[69]作为算法的实验平台，实验数据集和相关参数设置如表 4-2 所示，本章实验视频轨迹使用了《大巴克兔电影》，时长为 597s，一共 199 个视频分片，每个视频分片时长 3s，视频可供选择的比特率有 10 种，编码比特率集合为{230,331,477,688,991,1427,2056,2962,5027,6000},单位为 kbps。实验采取的最大缓冲区容量为 25s，并且通过结合算法理论和实践测试，最终设置相关参数B_{low}=0.6, B_{high}=0.9。

表 4-2 实验数据集和实验参数

Table 4-2 Experimental datasets and parameters

名称	数据集和实验参数
视频比特	{230,331,477,688,991,1427,2056,2962,5027,6000}Kbps
B_{low}	0.6
B_{high}	0.9
切片时间	3s
缓冲区大	25s
网络轨迹	HSPDA

为了评估本章所提算法在现实网络条件下的性能，本章使用了在 Norway 收集的 HSDPA[70]数据集中火车和汽车的移动网络轨迹。为了匹配视频下载完成所需要的时间，本章利用滑动窗口将 HSDPA 中火车和汽车的网络轨迹分别切成 500 个网络轨迹，每个轨迹持续 632 秒。在本章实验环境下，将融合了本章所提算法的 BOLA-E 算法和 THROUGHPUT 算法与原始算法进行性能比较。在 4.4.3 节中展示本章所提出算法在模拟平台下的性能表现。

4.5.2 评价指标

为了评估本章所提算法的性能，本章采用了 X. Yin 等[27]提出的 QoE 评价模型，该模型将影响用户 QoE 的主要因素归纳为视频的质量，视频播放的平滑度，视频播放过程中的重新缓冲事件，然后用不同的权重进行组合，具体表现形式如(4-6)所示：

$$QoE = \sum_{i=1}^{n} q(R_i) - \mu . \sum_{i=1}^{i} T_i - \sum_{i=1}^{n-1} |q(R_{i+1} - q(R_i)| \qquad (4-6)$$

n表示视频切片数；R_i是第切片视频切片的比特率，$q(R_i)$表示视频切片的

效用值，该效用值表示用户感知的质量。T_i表示以比特率R_i下载第 i 切片所产生的重新缓冲时间为；μ 是权重，它决定了重新缓冲事件的影响程度，μ 值越大表明重新缓冲带来的影响越大。$q(R_{i+1} - q(R_i)$表示视频播放的相邻视频切片播放用户感知的质量差，差值之和用来表示视频播放的平滑度，值越大，说明平滑度越低，对 QoE 造成的损失就越大。视频质量越大更能提高用户的 QoE,而重新缓冲和低平滑度都会给用户的 QoE 带来负面影响，会降低用户的 QoE。为了更直观的感受 QoE 的提升，本章使用了用户感知质量的线性表示，即线性表示$q(R_i) = R_i$。同时选取了合适的权重 μ：μ=4300。

为了评估算法的性能，将所提算法分别融合到基于缓冲区的 ABR 算法 BOLA-E 和基于带宽的 ABR 算法 THROUGHPUT 中。评价指标包括平均 QoE 和平均比特率、平均比特率切换、平均重新缓冲率和平均重新缓冲事件组成。

其中平均 QoE 表示本章数据集下 QoE 的均值，其值越大，表明算法性能越好。平均比特率表示本章数据集下播放视频总比特率的均值，其值越大表示视频质量越高，视频越清晰。平均比特率切换表示本章数据集下播放视频的平均平滑度，其值越大，表示视频播放越平滑。重新缓冲率表示在播放视频的卡顿时间占总播放时间的百分比，重新缓冲事件表示播放视频发生卡顿的次数。

4.5.3 实验结果分析

图4-4、图4-5和图4-6分别显示了每个算法在本章数据集下的平均比特率，平均 QoE 以及平均比特率切换。这些结果表明融合了替换策略的自适应算法在平均比特率，平均 QoE 以及平均平滑度上所表现的性能明显优于原本算法，具体而言，在平均 QoE 方面，融合了替换策略的 BOLA-E 算法的平均 QoE 较原来相比提升了 11.6%，融合了替换策略的 THROUGHPUT 算法的平均 QoE 较 THROUGHPUT 算法提升了 17%。在平均比特率方面，融合了替换策略的 BOLA-E 算法的平均比特率较 BOLA-E 算法提升了 7%，融合了替换策略的 THROUGHPUT 算法的平均比特率较 THROUGHPUT 算法提升了 13%。在平滑度方面，融合了替换策略的 BOLA-E 算法的平均平滑度较 BOLA-E 算法提升了 10%，融合了替换策略的 THROUGHPUT 算法的平均平滑度较 THROUGHPUT 算法提升了 37%。

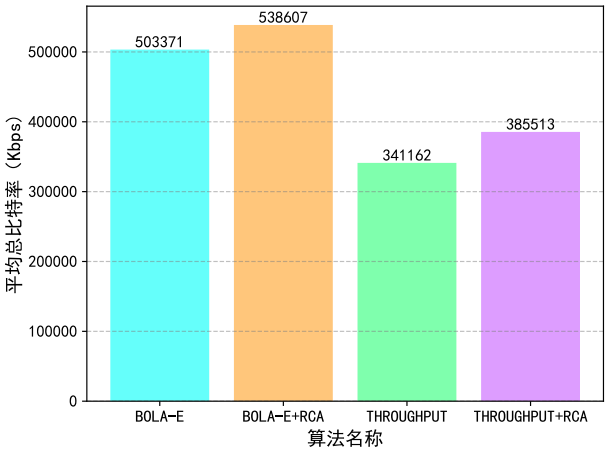


图 4-4 各算法平均总比特率

Figure 4-4 The average total bit rate of each algorithm

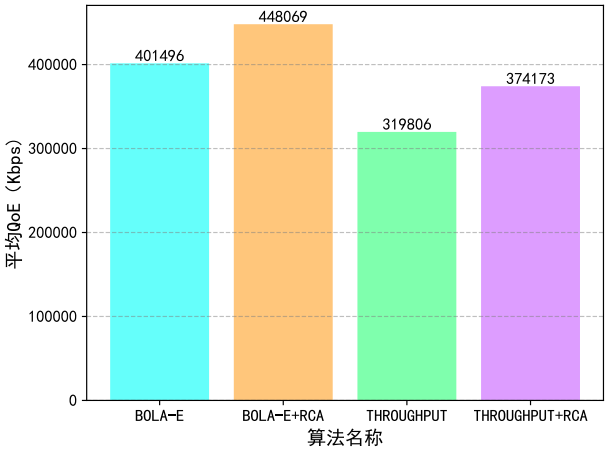


图 4-5 各算法平均 QoE

Figure 4-5 Average QoE of each algorithm

图 4-4 和图 4-5，发现在融合了替换算法之后，视频的平均 QoE 和平均比特率都有一定程度的上升，说明本章的替换算法在网络环境频繁波动情况下，能够有效的应用。

增加了替换的 THROUGHPUT 与 BOLA-E 相比，即使增加了替换，在平均比特率和平均 QoE 方面表现的性能仍略低。这是因为在网络环境频繁剧烈波动情况下，基于带宽自适应算法本身的缺点，不会选择超过当前带宽的比特率，同时视频比特率波动大，导致视频整体 QoE 较低，视频平均比特率较低。

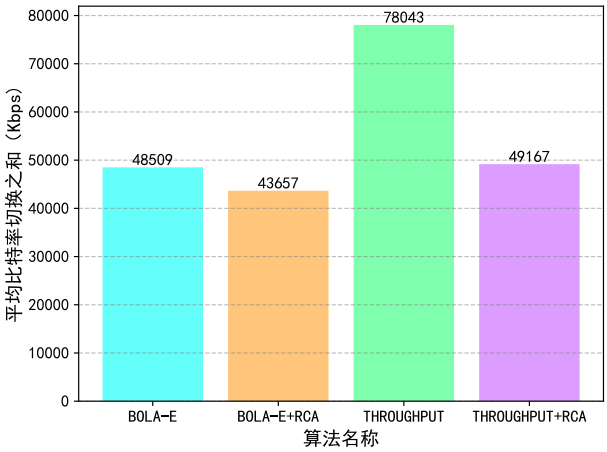


图 4-6 各算法平均比特率切换之和

Figure 4-6 The sum of the average bit rate switching of each algorithm

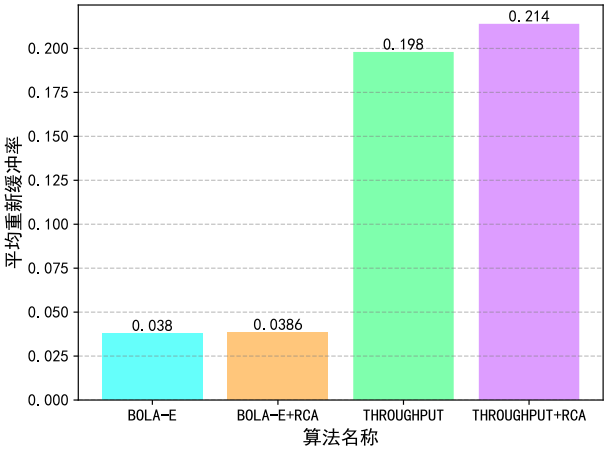


图 4-7 各算法平均重新缓冲率

Figure 4-7 Average rebuffering rate of each algorithm

图 4-6 中，本章发现现有的 ABR 算法在融合了本章提出的替换算法后，基于带宽的 ABR 算法在视频平滑度方面有明显提高，原因是网络带宽频繁波动的情况下，视频切片比特率波动大，总体比特率改变较大，视频平滑度较低。相比在基于带宽的 ABR 算法 BOLA-E 融合本章的算法后平滑度性能提升不是很明显，本章认为这是 BOLA-E 算法中占位符算法提供的优点所致，在 BOLA-E 中，占位符算法能够有效减少视频比特率波动。

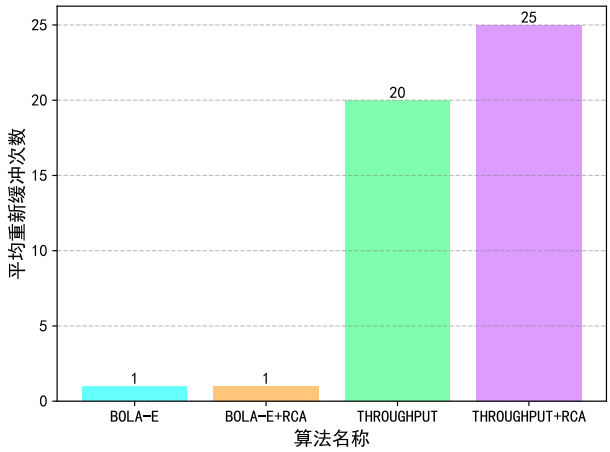


图 4-8 各算法平均重新缓冲次数

Figure 4-8 The average number of rebuffering times for each algorithm

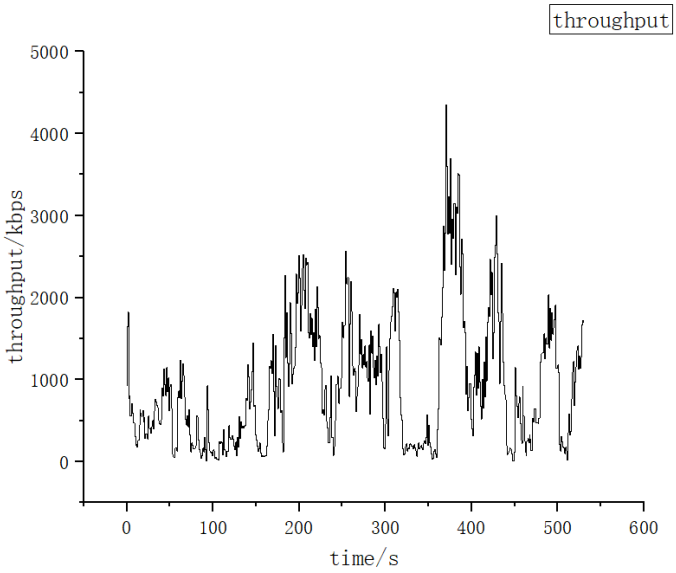


图 4-9 网络轨迹

Figure 4-9 network trajectory

图 4-7 和图 4-8 显示了在添加替换后，ABR 算法在重新缓冲的变化，本章发现总体来说，添加替换对重缓冲的影响较小。这是因为本章的替换假设当前的替换不会立即导致重缓冲。尽管影响不大，但对于吞吐量算法，由于重缓冲事件可能在替换后发生，替换并不增加缓冲，且在持续低带宽时缓冲严重耗尽，所以在添加替换后，重缓冲事件略有增加。本章的替换在一定程度上加速了缓冲消耗，从而导致重新缓冲。本章绘制了 HSPDA 数据集的部分网络轨迹，如图 4-9 所示。网络轨迹中有多个持续低带宽区域，这导致了重缓冲。基于带宽的

ABR 算法 THROUGHPUT 和融合了本章算法的 THROUGHPUT 算法的重缓冲事件明显高于 BOLA-E 和融合了本章算法的 BOLA-E，这是因为基于带宽的自适应算法没有考虑缓冲因素，而在持续低带宽期间，基于带宽的自适应算法缓冲被耗尽，从而大大增加了重缓冲，而基于缓冲的 ABR 算法考虑了缓冲因素，因此有较少的重缓冲事件和低概率。

4.6 本章小结

本章首先描述了无线流媒体 QoE 的影响因素，然后阐述了本章研究的目的。通过分析现有 ABR 算法的不足，针对问题，提出了一种基于 QoE 感知的 ABR 算法。该算法可以适用于无线环境下。本章描述了该算法的逻辑，主要是在已有的自适应算法的基础上，增加本章的替换策略，在适当的实际以更高比特率的视频切片替换缓冲去区中尚未播放的视频切片。随后本章描述了如何根据替换策略实现视频切片替换。最后本章在模拟平台 sabre 上评估了算法的性能，与 BOLA-E 和 THROUGHPUT 相比，融合了替换的 ABR 算法，在平均 QoE 上分别提高了 11.6%和 17%。在平均比特率方面，融合了替换策略的 BOLA-E 算法的平均比特率较 BOLA-E 算法提升了 7%，融合了替换策略的 THROUGHPUT 算法的平均比特率较 THROUGHPUT 算法提升了 13%。即使在网络环境波动很大的情况下，增加了替换的 ABR 算法在重新缓冲方面没有明显增加。

第 5 章 基于 COMA 的多客户端流媒体 QoE 优化算法

本章针对现有的 ABR 算法在多客户端竞争同一网络带宽资源时，带宽分配不均匀且用户 QoE 较低的问题。提出了一种基于 COMA 的多客户端流媒体 QoE 优化算法。算法的目标是在保证各用户不卡顿的基础上，最大化用户的 QoE，同时维护了各客户端之间的公平性。本章首先进行了问题描述，接着对无线环境下多客户端竞争同一网络带宽资源场景中的比特率决策问题进行 Dec-POMDP 建模，然后基于多智能体深度强化学习 COMA 方法进行求解，客户端根据训练好的智能体进行比特率选择。最后在仿真平台上，对所提算法进行实验验证并进行结果分析。

5.1 问题描述

图 5-1 显示了多客户端竞争同一网络带宽资源的场景。当多客户端竞争同一网络带宽资源时，不同用户之间存在带宽分配的公平性问题。如果带宽分配不均匀，可能会造成部分用户以普清播放，甚至卡顿，而少部分用户却能以高清播放。同时在资源竞争过程中，带宽资源的分配不是固定不变的，这导致了用户的实际可用带宽波动大，从而造成了比特率切换频繁等现象。这种场景下，会严重影响用户的 QoE。

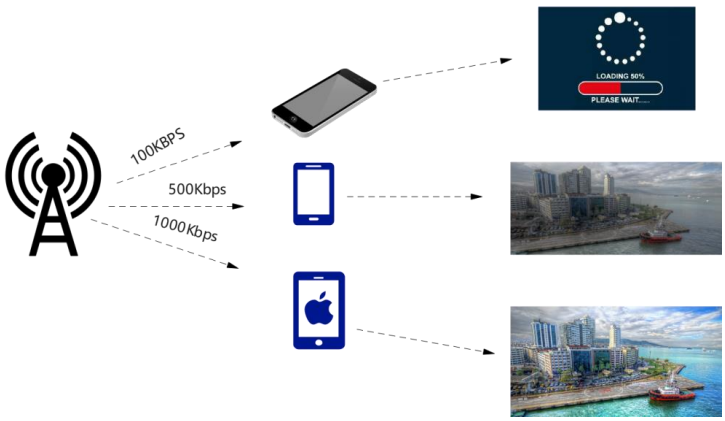


图 5-1 多客户端竞争同一网络带宽资源的场景

Figure 5-1 Multiple clients compete for bandwidth resources on the same network

5.2 问题建模

本节将针对无线环境下多客户竞争同一网络带宽资源时比特率决策的场景进行问题建模。主要包括两部分数学建模和 Dec-POMDP 建模。

5.2.1 数学建模

依据 DASH 技术的特点，该场景的数学建模包括以下特征：1.每当上一个视频切片下载完成后，客户端的 ABR 算法会立即决策下一个视频切片的比特率；2. 可选比特率集合应该是离散且有限的数值集合；3.客户端的视频播放器缓冲区大小应该有限，且当缓冲区满时，客户端不再进行视频切片下载；4.客户端 ABR 算法无法决定带宽的分配；5.当视频播放器缓冲区中为空时，播放器将发生重新缓冲，即发生卡顿，此时必须等缓冲区中已经下载了一个完整视频切片才能继续播放，重新缓冲的时间被记为卡顿时间。

为了便于建模而不失一般性，本章假设所有客户端在同一时刻播放同一视频，且在该视频下载完成后不再下载其他视频。本章算法的系统目标在保证每个用户不卡顿的基础上，尽可能提高所有用户的 QoE，同时，保证用户之间的 QoE 公平性。只要发生卡顿，该用户获得的 QoE 为 0。当客户端播放器不发生卡顿的情况下，用户的 QoE 由所有视频切片比特率和比特率切换所决定；根据所述问题的特点和假设条件，将问题数学化，客户端集合$C = \{c_1, c_2, \ldots, c_N\}$，N 表示客户端的数量，$c_i$表示每个客户端。视频切片集合$V = \{v_1, v_2, \ldots v_M\}$的视频，M 表示视频切片数量，$v_j$表示每个视频切片。每个视频切片的时间为 Δt，客户端播放器可以选择的比特率集合为 R。对于每个客户端c_i中的 ABR 算法需要进行如式(5-1)所示的比特率决策：

$$\begin{aligned} &R_{c_i} = \{r_{c_i}^{v_1}, r_{c_i}^{v_2}, \ldots, r_{c_i}^{v_j}, \ldots, r_{c_i}^{v_M}\} \\ &\text{st} \ \ r_{c_i}^{v_j} \in R, \forall j \in \{1,2,\ldots,M\} \end{aligned} \tag{5-1}$$

其中，$r_{c_i}^{v_j}$表示客户端 c_i下载第 j 个视频切片v_j时选择的比特率，$r_{c_i}^{v_j}$包含在 R 中。本模型的系统目标是在保证每个用户不卡顿的基础上，尽可能提高所有用户的 QoE，同时保证用户之间的 QoE 公平性。因此，优化该系统目标与该系统中所有客户端用户观看视频所获得的 QoE、客户端卡顿时间以及各客户端用户观看视频获得的 QoE 的标准差相关。客户端c_i中用户观看完整视频所获得的

QoE 值QoE_{c_i}如公式(5-2)所示。

$$QoE_{c_i} = f(R_{c_i}) \cdot \{\sum_{j=1}^{n} q(r_{c_i}^{v_j}) - \sum_{j=1}^{n-1} |q(r_{c_i}^{v_{j+1}} - q(r_{c_i}^{v_j})|\} \quad (5-2)$$

其中$q(r_{c_i}^{v_j})$表示视频质量所带来的 QoE，$|q(r_{c_i}^{v_{j+1}} - q(r_{c_i}^{v_j})|$质量切换带来的惩罚项。$f(R_{c_i})$是客户端$c_i$在播放视频过程中总卡顿时间$R_i$的指示函数。具体形式如公式(5-3)所示：

$$f(R_i) = \begin{cases} 1 & R_{c_i} = 0 \\ 0 & R_{c_i} > 0 \end{cases} \quad (5-3)$$

即一旦发生卡顿，该客户端用户的 QoE 值为 0。结合系统目标，本系统的优化目标如公式 (5-4)所示：

$$Maximize \quad \alpha \cdot \sum_{i=1}^{N} Q_{c_i} - \beta \cdot \sigma(Q_c) \quad (5-4)$$

其中，α、β 为权重系数，通过调节用以平衡系统的总 QoE 和个客户端 QoE 标准差。$\sigma(Q_c)$表示各客户端 QoE 的标准差，计算方式如公式(5-5)所示：

$$\sigma(Q_c) = \sqrt{\frac{\sum_{i=1}^{N} (Q_{c_i} - \frac{1}{N}\sum_{i=1}^{N} Q_{c_i})^2}{N-1}} \quad (5-5)$$

在以上模型中，每个客户端的 ABR 算法无法获取其他客户端的信息，仅能够根据本地信息（客户端缓冲区大小、历史下载信息）进行比特率决策。其中历史信息包括过去一段时间下载视频的比特率，下载视频切片的大小，下载视频切片的时间等。但为了实现系统目标，每个客户端的 ABR 算法需要协同进行比特率决策，以便实现带宽资源的合理分配，同时最大化用户的 QoE。上述系统目标可以简单理解为多客户端共同进行比特率决策从而实现整个系统的目标因此本文考虑将上述模型进一步进行 Dec-POMDP 建模。

5.2.2 Dec-POMDP 建模

在多客户端竞争同一网络带宽的场景中，首先为每个客户端c_i分配一个智能体a_i进行动作选择。视频播放器根据智能体的动作，选择相应的比特率进行视频切片下载。由于系统的目的是最大化所有用户的 QoE，因此智能体之间是纯合作关系。在 5.3.1.1 小节的数学模型中，每个客户端c_i中的 ABR 算法在上一个视频切片下载完成后，下一个视频切片下载开始时进行比特率决策。由于每

个客户端选择的比特率和下载速度可能不同，每个客户端进行相同视频切片比特率决策的时刻也可能不同。Dec-POMDP 是一种分布化的框架，其中每个智能体都只能观察到部分环境状态，而不能完全观察到整个系统的状态。每个智能体都需要独立地制定决策策略，并且必须考虑到其他智能体的行为和观察。但是，在 Dec-POMDP 建模中，所有智能体需要同时进行动作选择。因此本文采取固定时间间隔决策的方式来实现同时决策。即所有客户端播放器在开始播放视频后，每经过 ΔT 时间，每个客户端c_i的智能体都会进行一次动作选择。客户端播放器根据智能体选择的动作，选择相对应的比特率。因此，针对该问题的多客户端协同决策的 Dec-POMDP 建模如图 5-2 所示。

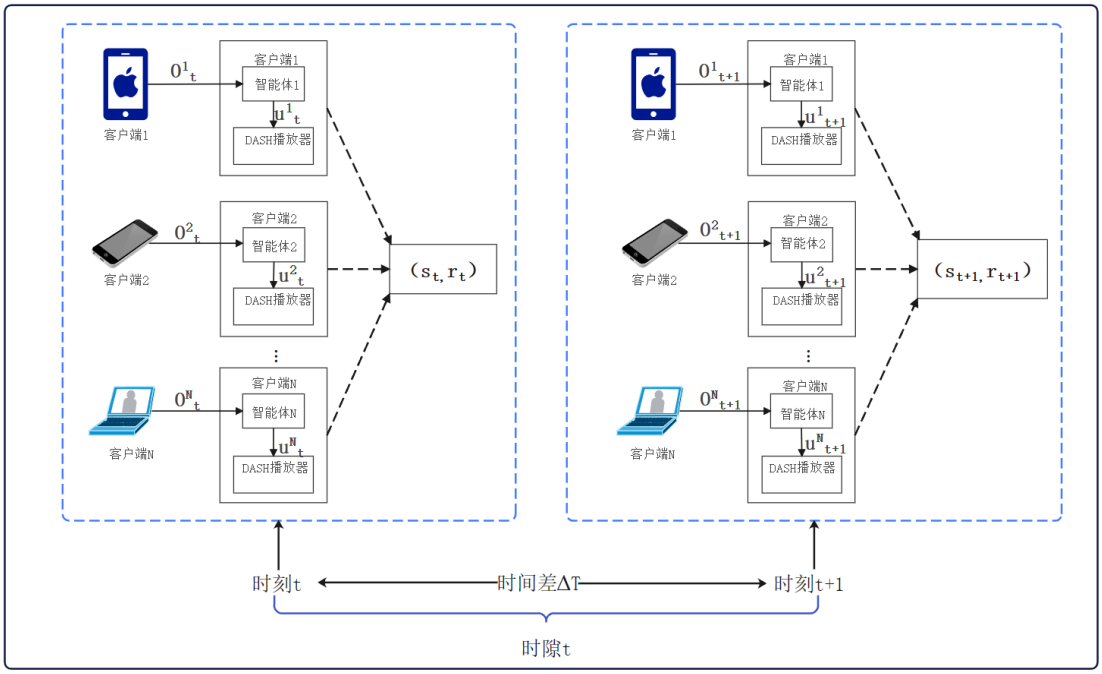


图 5-2 多客户端协同决策的 Dec-POMDP 建模

Figure 5-2 Dec-POMDP modeling for multi-client collaborative decision-making

在第 t 时刻，各客户端c_i中的智能体a_i根据当前时刻的本地观察o_t^i选择动作u_t^i。t 时刻的全局状态记为s_t,它包括所有智能体在 t 时刻的本地观察和部分全局信息（单智能体无法获取的信息）。在接下来的 ΔT 时间内，各客户端播放器根据智能体选择的动作，选择相应的比特率进行视频切片下载。在 t+1 时刻系统返回所有智能体在 t 时刻的联合动作$\{u_t^1, u_t^2, \ldots, u_t^N\}$的全局奖励$r_t$。t+1 时刻每个智能体$a_i$的新的本地观察为$o_{t+1}^i$，此时全局状态记为$s_{t+1}$。重复上述过程，直到所

有客户端完成所有视频切片的下载。在下载过程中，客户端播放器根据智能体选择的动作进行比特率决策时存在以下几种情况：

1）在 t 到 t+1 时间内，客户端播放器需要进行多次比特率决策。这种情况出现的原因可能是 t 时刻智能体选择的动作对应的比特率较低或者在此 ΔT 时间内客户端分配的带宽较高，导致每个视频切片的下载时间过短，从而在 ΔT 时间内需要进行多次下载决策。

2）在 t 到 t+1 时间内，客户端播放器不需要进行比特率决策。这种情况出现的原因可能包括一下三点：i）t-1 时刻智能体选择的动作对应的比特率较高或者在此 ΔT 时间内客户端分配的带宽较低，这都导致视频切片的下载时间过长，在 t 时刻之前开始下载的视频切片直到 t+1 时刻仍未完成，导致即使智能体在 t 时刻进行了动作选择，而客户端在在 t 到 t+1 时间内不需要进行比特率决策。ii）t 时刻客户端播放缓冲区已满，这导致视频无法继续下载，因此和当前客户端播放器已经完成了所有视频切片的下载。导致即使智能体选择了动作，而客户端播放器不需要进行比特率选择。iii）当前客户端播放器已经完成了所有视频切片的下载，但其他客户端没有完成，此时该客户端的智能体选择无效动作，客户端播放器不需要进行比特率选择。

5.3 算法设计

5.3.1COMA 算法介绍

COMA 算法是一种基于集中式训练分布式执行(CTDE)框架的多智能体深度强化学习算法，由 Jakob N. Foerster 等人在 2018 年提出。它旨在解决多智能体系统中的信用分配问题,即如何将全局奖励合理地分配给各个智能体,以促进它们的合作与协调。因此，COMA 算法能够对纯合作式的 Dec-POMDP 问题就行求解,这也是本文选取 COMA 算法求解的原因。COMA 算法主要基于强化学习 Actor-Critic 方法发展而来，核心思想是为每个智能体引入一个共同 Critic 网络,用于估计该智能体的行为对全局奖励的贡献。图 5-3 显示了 COMA 中分布式的 Actor、环境和集中式的 Critic 之间的交互，图 5-4(a)显示了 Actor 网络的具体结构，图 5-4(b)显示了 Critic 网络的具体结构。其中s_t表示 t 时刻的全局状态，它主要由 t 时刻所有智能体的本地观察o_t^a和部分全局信息组成。u_t^a表示智能体a在 t 时刻选择的动作，$\boldsymbol{u}_t^{-a}$表示除智能体a以外其他智能体在 t 时刻的联合动作。

$\boldsymbol{u}$表示 t 时刻所有智能体的联合动作，$\boldsymbol{u}_{t-1}$表示 t-1 时刻所有智能体的联合动作，r_t表示t时刻联合动作u的全局奖励，$h^a{}_{t-1}$表示上一个时刻的隐藏信息，$|U|$表示动作空间的大小。智能 COMA 算法使用多个分布式的 Actor 网络对每个智能体的行为策略π进行迭代更新，使用一个集中式的 Critic 网络评估每个智能体的行为对全局奖励的贡献。

算法原文中指出，如果在多智能体合作场景中使用传统 Actor-Critic 方法中 Critic 网络的 TD 误差对每个智能体a的策略梯度进行迭代更新（如公式(5-6)所示）很难评估单个智能体对整体的贡献，容易造成智能体“偷懒”的现象。

$$g = \nabla_{\theta^\pi} log\pi(u|\tau_t^a)\big(r + \gamma V(s_{t+1}) - V(s_t)\big) \qquad (5-6)$$

因此，COMA 算法提出了一种反事实基线（Conter factual Baseline）的方法，通过计算每个智能体的优势函数，来评估每个智能体的贡献。Critic 网络根据输入 $(\boldsymbol{u}_t^{-a}, s_t, o_t^a, a, \boldsymbol{u}_{t-1})$计算智能体 a 采取每个动作的 Q 值。Critic 网络的输出为 t 时刻每个智能体 a 的优势函数值A_t^a。优势函数的计算方式如式如式(5-7)所示。

$$A^a(s, \boldsymbol{u}) = Q(s, \boldsymbol{u}) - \sum_{u'^a} \pi^a(u'^a|\tau^a) Q\big(s, (u^{-a}, u'^a)\big) \qquad (5-7)$$

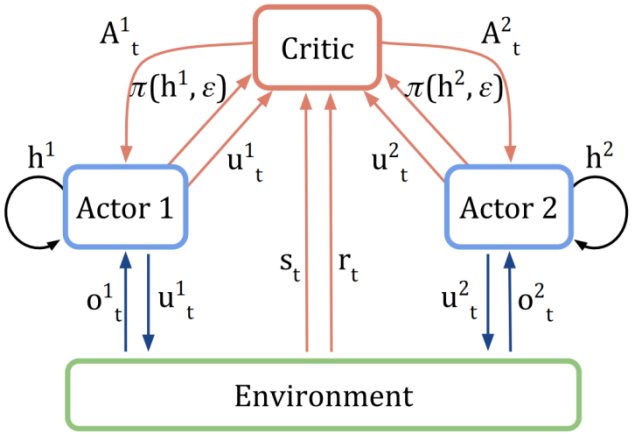


图 5-3 COMA 中分布的智能体、环境和集中式的 Actor-Critic 之间的交互[56]

Figue5-3 information flow between the decentralised actors, the environment and the centralised critic in COMA[56]

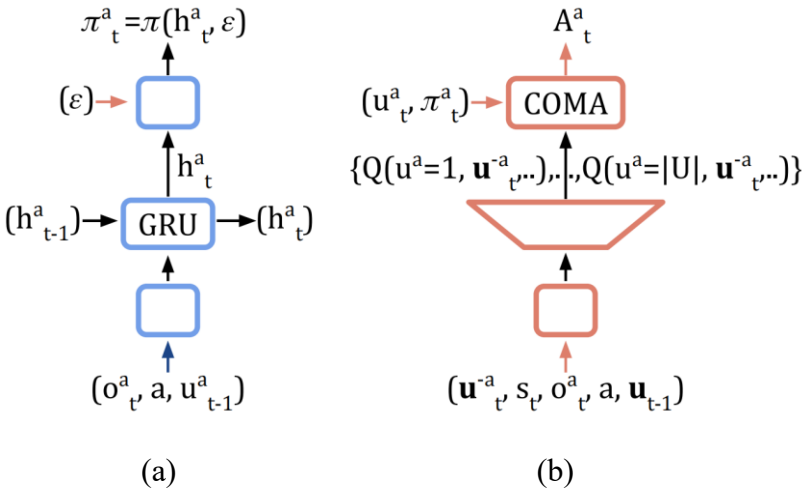


图 5-4 网络结构[56]。 (a) Actor 网络结构；(b)Critic 网络结构

Figure 5-4 Network structure[56]. (a) Actor network structure; (b) Critic network structure

其中，$Q(s,\boldsymbol{u})$表示 t 时刻所有智能体联合动作$\boldsymbol{u}$的状态-动作价值函数。u'^a表示动作空间中除去动作 u^a的动作集合。$A^a(s,u)$为每个智能体计算出各自的基线。在算法的训练过程中，Critic 网络通过所有智能体执行动作的经验数据(u,o,s,r)进行更新，Actor 根据 Critic 网络输出的优势函数进行更新。同时相比其他多智能体强化学习算法,COMA 算法能更快收敛到较优策略。综上所述 COMA 算法在解决纯合作场景的 Dec-POMDP 问题具有明显的优势，因此本文基于 COMA 算法，提出了适用于无线环境下多客户端竞争同一网络资源场景的多客户端流媒体 QoE 优化算法。

5.3.2 算法框架

如图 5-5 所示，为本文基于 COMA 的多客户端流媒体 QoE 优化算法的框架图。每个客户端都被分配一个智能体，在 t 时刻每个智能体a根据 Actor 网络进行动作选择。Actor 网络的输入是本地观察o_t^a，输出是动作u_t^a。客户端播放器根据智能体抉择的动作u_t^a选择相对应的比特率。每个智能体在选择动作后，会在ΔT时间后获得一个全局奖励r_t，在此ΔT时间内客户端播放器以同一比特率下载视频切片。在训练过程中，全局状态s_t由所有智能体的本地观察o_t和部分全局信息组成。在每轮训练的开始，会进行多次视频播放仿真，在每次仿真中各个智能体根据 Actor 网络进行连续动作选择，播放器根据动作下载对应比特率的视频切片，直到所有视频切片下载完成，该次仿真结束。将视频播放仿真过程中

各个智能体选择的动作u、各智能体的本地观察o、全局状态s和获得的全局奖励r以(u,o,s,r)的形式存入经验池中。在多次视频播放仿真实验结束后，目标 Critic 网络根据经验池中的经验信息(u,o,s,r)对 Critic 网络进行更新，更新后的 Critic 网络输出 COMA 优势函数并指导 Actor 网络进行更新，当 Critic 网络和 Actor 网络更新完成之后将进入下一轮的训练。在视频播放仿真的实验过程中 Critic 网络和 Actor 网络都不会更新。在使用 COMA 方法对 Dec-POMDP 问题求解时，观察空间、动作空间、全局状态空间、和全局奖励函数的设计至关重要。因此，观察空间的设计应该包含本地影响比特率抉择的因素。动作空间的设计应该具有指导客户端比特率选择的实际意义。全局状态空间的设计应该不仅包括所有智能体的本地观察，同时包含全局信息，该信息是其他智能体无法获取的。奖励函数的设计应该尽可能体现系统的目标，即在该场景中，各个智能体之间相互协作，共同实现最大化用户 QoE。

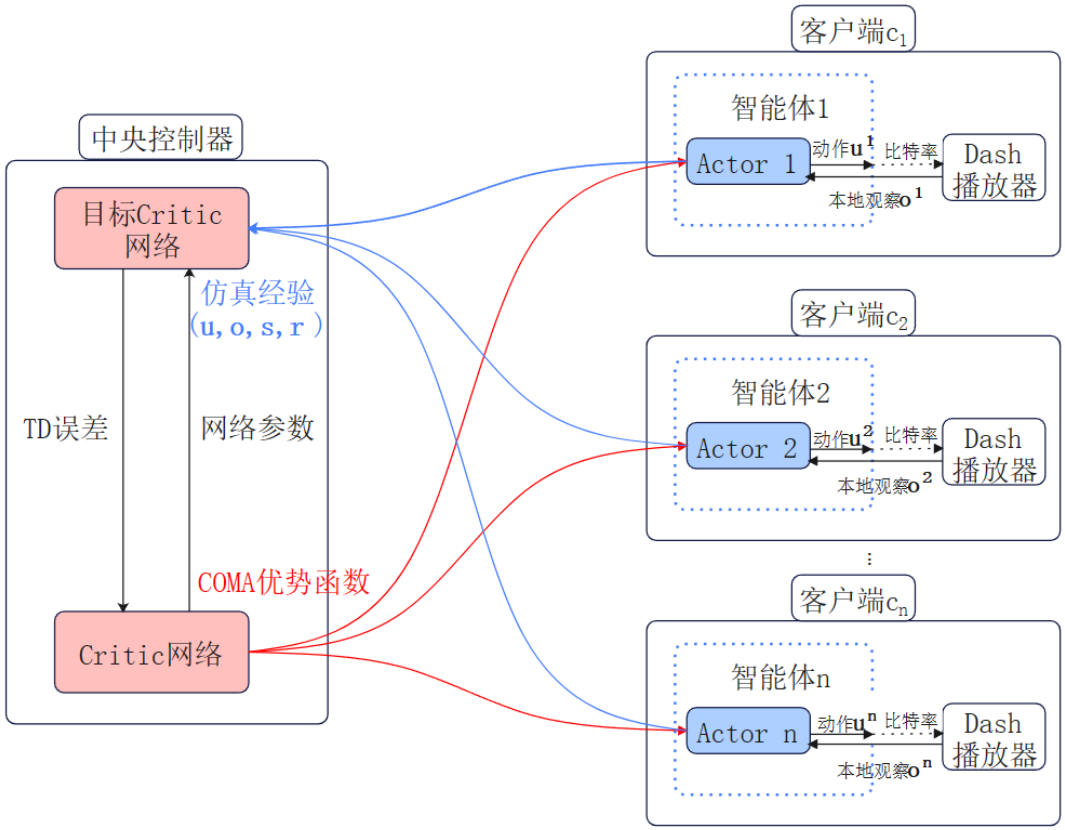


图 5-5 基于 COMA 的多客户端流媒体 QoE 优化算法框架

Figure 5-5 Multiclient streaming media QoE optimization algorithm framework based on COMA

5.3.3 观察空间、全局状态空间、动作空间设计

观察空间是单个智能体在决策时看到的局部信息，在本文的算法框架中，训练好的智能体仅根据本地的观察进行动作选择，无需看到全局信息。因此，观察空间的设计应该与基于强化学习的单客户端 ABR 算法的状态空间相匹配。Pensieve[47]算法是基于强化学习的单客户端 ABR 算法中典型的算法，且性能优异。因此，本文以 Pensieve 算法的状态空间作为本文算法观察空间设计的参考。其中包括过去 K 个视频切片的吞吐量估计、过去 K 个视频切片的下载时间、下一个视频切片的可选大小、当前播放器缓冲区大小、上一个视频切片的比特率和尚未下载的视频切片数量。在本文的算法框架中，智能体 Actor 网络的输入是本地观察，输出是选择的动作，即客户端根据选择的动作选择相对应的比特率。在 Pensieve 算法的状态空间中之所以包括过去 K 个视频切片的吞吐量估计和过去 K 个视频切片的下载时间的目的是估算当前的吞吐量。因此，本章考虑将当前播放器缓冲区大小、上一个视频切片的比特率以及当前的吞吐量估计作为观察空间的一部分。

此外，根据本章 5.3.1 节中的 Dec-POMDP 建模，所有智能体每经过 ΔT 进行一次动作选择。客户端 ΔT 时间内进行恒定比特率下载，为了保证视频播放不卡顿，需要理解 ΔT 时间内下载的视频切片数量。综上，以$o_t^i = \{b_t^i, T_t^i, r_t^i, x_t^i\}$。作为 t 时刻智能体$a_i$的本地观察，其中$B_t^i$为 t 时刻客户端$c_i$的缓冲区大小，$T_t^i$为 t 时刻客户端$c_i$的吞吐量估计，其计算方式为过去 5 个视频切片的文件总大小与下载这 5 个视频切片花费时间的，r_t^i为客户端上一次的比特率选择；x_t^i为上一个 ΔT 时间内下载的视频切片数量。

全局状态包含了环境中所有智能体的状态信息和可能的环境变量，它为算法提供了完整的环境视图，使算法能够更好地理解每个智能体的行为如何影响整个环境。COMA 算法利用全局状态来计算每个智能体行为的对比差异。因此，全局状态的设计必须包含所有智能体的状态，同时包含所有智能体的联合信息，该信息用于多智能体中央控制器，在训练过程中指导单智能体进行决策。因此，全局状态应为$s_t = \{\{o_t^1, \ldots, o_t^i, \ldots, o_t^N\}, p_t^1, p_t^2, \ldots, p_t^N\}$，其中$p_t^0$到$p_t^N$表示，每个客户端$c_i$在 t 时刻是否为已经完成了整个视频的下载。$p_t^a$为 1 表示智能体 a 对应的

客户端已经完成了整个视频的下载，p_t^a为 0 表示尚未完成整个视频的下载。该环境变量在训练阶段能够被中央控制器利用，从而间接指导单智能体的行为。例如，当某些智能体对应的客户端已经完成下载时，中央控制器根据全局状态会评估智能体的行为，从而让单智能体仅通过本地状态而感受到环境中，竞争的客户端在减少，从而选择更高的比特率，并且不会发生卡顿。

智能体的动作空间表示为$u = \{a_1, a_2, \ldots, a_{|R|}, a_0\}$，其中$a_1$到$a_{|R|}$对应着每一个可以选择的视频比特率。$|R|$表示视频比特率集合的大小，$a_0$表示不需要执行的动作。它表示当前客户端已经完成了所有视频切片的下载。

5.3.4 全局奖励函数设计

为了实现系统目标，全局奖励函数的设计应该保证尽可能每个客户端不发生卡顿。如果发生卡顿，该用户的 QoE 为 0。因此，设计了如式(5-8)和式(5-9)所示的全局奖励函数:

$$r_t = \sum_{i=1}^{N} Q_t^i - \eta \cdot \sigma(Q_t), Q_t = \{Q_t^1, \ldots, Q_t^N\} \tag{5-8}$$

$$Q_t^i = \begin{cases} r_t^i \cdot x_t^i - u\left|r_t^i - r_{t-1}^i\right| & l_t^i \geq \Delta T \\ 0 & l_t^i < \Delta T \end{cases} \tag{5-9}$$

其中x_t^i 表示ΔT时间内下载的视频切片数量， l_t^i表示 t 时刻客户端c_i缓冲区中尚未播放的视频时长B_t^i与客户端c_i在ΔT时间内下载的视频切片时长之和，即$l_t^i = B_t^i + x_t^i \cdot \Delta t$, 其中$\Delta t$表示每个视频切片的时长。当 $l_t^i \geq \Delta T$时，意味着在ΔT时间内，客户端缓冲区中有足够的视频支持播放，该时间内，客户端播放视频不会卡顿。因此将$r_t^i \cdot x_t^i - u\left|r_t^i - r_{t-1}^i\right|$作为每个智能体在执行完动作后获得的奖励，使得智能体能够在保证客户端视频播放不卡顿的最低前提下，尽可能提高选择的比特率的同时减小比特率切换；当$l_t^i < \Delta T$时，表明客户端在播放过程中，出现缓冲区耗尽，视频重新缓冲的的情况。此时客户端播放会发生卡顿。可能原因是 t 时刻选择的比特率r_t^i较高，而没有足够的带宽资源支持下载，从而导致下载缓慢，缓冲区中视频已经全部播放完。在系统目标中，卡顿是不允许的，因此奖励为 0。

表 5-1 基于 COMA 的多客户端流媒体 QoE 优化算法

Table 5-1 QoE optimization algorithm for multi-client streaming media based on COMA

算法 5-1 基于 COMA 的多客户端流媒体 QoE 优化算法

1:初始化 $\theta_1^c, \hat{\theta}_1^c, \theta^\pi$ //初始化智能体 Actor 网络参数θ^π，Critic 网络参数θ_1^c

2: ***for*** $e = 1$ ***to*** E ***do*** target Critic 网络参数$\hat{\theta}_1^c$

3: 清空经验缓冲区 //初始化一个空的缓冲池

4: ***for*** $e_c = 1$ ***to*** $BatchSize/n$ ***do***

5: $s_1 =$ 初始状态, $t = 0, h_0^a = \mathbf{0}$ //为每个智能体a初始化

6: **while** $s_t \neq$ terminal and $t < T$ **do** //整个视频尚未完全下载

7: $t = t + 1$

8: ***for*** 每个智能体a **do**

9: $h_t^a = \text{Actor}(o_t^a, h_{t-1}^a, u_{t-1}^a, a, u; \theta_i)$

10: 根据策略$\pi\left(h_t^a, \epsilon(e)\right)$执行动作$u_t^a$

11: 获取奖励r_t和下一个状态s_{t+1}

12: ***end for***

13: 将每轮经验添加到缓冲区

14: 收集经验池中的仿真数据组成该轮经验

15: ***end for***

16: ***for*** t = 1 **to** T **do** //通过单批次并行训练所有智能体

17: 使用 RNN 处理状态(states)、动作(actions)、和奖励(rewards)

18: 使用$\hat{\theta}_i^c$计算$TD(\lambda)$目标y_t^a

19: ***end for***

20: ***for*** t = T **down to** 1 **do**

21: $\Delta Q_t^a = y_t^a - Q\left(s_j^a, \mathbf{u}\right)$

22: $\Delta\theta^c = \nabla_{\theta^c}(\Delta Q_t^a)^2$ //计算 Critic 网络梯度

23: $\theta_{i+1}^c = \theta_i^c - \alpha\Delta\theta^c$ //更新 Critic 网络权重参数

24: 每 C 步重置$\hat{\theta}_i^c = \theta_i^c$ //更新目标 Critic 网络权重参数

25: ***end for***

26: ***for*** t = T **down to** 1 **do** //计算 COMA 优势函数

27: $A^a(s_t^a, \mathbf{u}) = Q(s_t^a, \mathbf{u}) - \sum_u Q(s_t^a, u, \mathbf{u}^{-a})\pi(u|h_t^a)$

28: $\Delta\theta^\pi = \Delta\theta^\pi + \nabla_{\theta^\pi}\log \pi(u|h_t^a)A^a(s_t^a, \mathbf{u})$ //计算 Actor 网络梯度

29: $\theta_{i+1}^\pi = \theta_i^\pi + \alpha\Delta\theta^\pi$ //更新 Actor 网络权重参数

30: ***end for***

31: ***for*** $t = 0$ ***to*** T ***do***

32: ***for*** 每个智能体a **do**

33: $h_t^a = \text{Actor}(o_t^a, h_{t-1}^a, u_{t-1}^a, a, u; \theta_i)$

34: 根据策略$\pi\left(h_t^a, \epsilon(e)\right)$执行动作$u_t^a$

35: 各客户端根据智能体动作选择比特率

36: 获取奖励r_t和下一个状态s_{t+1}

37: $t = t + 1$

38: ***end for***

39: ***end for*** //视频下载结束

40: ***end for***

5.3.5 算法流程

算法 5-1 展示了基于 COMA 的多客户端流媒体 QoE 优化算法。算法 4-15 行的作用是每轮开始时收集仿真经验，具体为在每轮实验过程中，多次执行仿真。每次仿真中系统每 ΔT 时间收集一条经验(u,o,s,r)，直到所有客户端都完成所有视频切片下载。将多次执行的仿真经验加入经验池，组成该轮仿真经验。如算法第 10 行所示，在收集仿真经验过程中，智能体根据 Actor 策略网络和 ε-greedy 贪心策略共同选择要执行的动作。使用 ε-greedy 贪心策略的目的是在训练过程中增加探索，让智能体能够在不断尝试新的行动中发现并学习到更优的策略，从而提高其在未知环境下的性能表现。此外，t 时刻所有智能体选择动作之后，不会立即获得全局奖励r_t，然是要等到 t+1 时刻才能够计算全局奖励。在每一轮经验收集完成后， 进行 Critic 网络和 Actor 网络的更新。算法第 20 到 23 行表示，Critic 网络根据TD误差进行更新。算法第 24 行表示， 每 C 次动作选择后，将 Critic 网络的参数赋值给目标 Critic 网络。算法第 26-30 行是利用公式 (5-10) 中的反事实基线方法对 Actor 网络进行更新。由于 COMA 算法是同策略的深度强化学习方法，因此 COMA 算法仅仅使用本轮经验进行更新，下一轮开始时，上一轮经验将会被清空。算法 31-39 行，表明智能体根据训练好的 Actor 网络进行动作选择，客户端根据智能体动作选择相对应的比特率，直到整个视频下载完成。

5.4 实验与分析

5.4.1 实验设置

为了验证本章所提算法的性能，使用 VideoStream-NS3[71]模拟平台构建了包括 3 个视频客户端以及 1 台视频服务器的模拟环境。其中模拟平台环境参数设置如表 5-2 所示。

5.4.2 训练参数设置

在模拟实验中训练使用的平台包括：PC 机配置为 Nvidia 1080ti 11G、32G 内存、Ubuntu 16.04 操作系统作为实验平台，Python3.9、Pytorch1.8.0 等开发工具。具体的训练参数设置如表 5-3 所示。

表 5-2 模拟环境参数设置

Table 5-2 parameters for the simulated environment

参数	参数值
竞争的网络带宽	10Mbps
客户端缓冲区大小	25s
所有客户端播放的视频长度	100s
智能体决策间隔	4s
视频可选比特率集合	{300，750，1200，1850，4300}Kbps
视频去切片时长	2s
所有客户端初始播放的比特率	300Kbps

表 5-3 训练参数设置

Table 5-3 training parameter settings

参数	参数值
Actor 网络学习率	5×10^{-4}
Critic 网络学习率	3×10^{-4}
衰减因子 γ	0.99
优化器类型	RMSP
梯度裁剪	5
训练轮数	4000
探索方式	$\varepsilon - greedy$
种子数量	4

5.4.3 评估标准

为了验证本章所提算法的性能，本章选择了三个多客户端 ABR 算法作为本章算法的对比算法，包括 BBA、FESTIVE、SFTM[72]和 FDASH[73]。

本章所提算法的评价指标包括总 QoE、各客户端 QoE 标准差、各客户平均比他率标准差、总卡顿时间、总比特率切换以及各客户端平均比特率之和。

其中总 QoE 表示所有客户端播放视频所获得的视频体验质量之和，值越高表明算法在 QoE 方面的性能越优越。QoE 标准差反映了算法的公平性，值越小表明算法越公平，所有客户端都公平使用带宽资源。总卡顿时间表示所有客户端播放视频过程中的重新缓冲时间之和。总比特率切换表明所有客户端在播放视频过程中，发生比特率切换的所有差值之和。各客户端平均比特率之和表示所有客户端选择的视频比特率的均值之和，值越高表明算法能够提供的视频整体质量越大。

5.4.4 训练结果

在实验设置的条件下，使用如表 5-3 所示的参数进行训练。将训练过程可视化，如图 5-6 到 5-9 所示。其中每张图像中的深蓝色实线是各个种子在该图像中相应指标的均值，淡蓝色区域表示相应指标在各个种子上的方差。图 5-6 显示了累计奖励随训练过程变化趋势，可以看出累计奖励先平缓上升，在 300 轮次后累计奖励逐渐下降为 0，并且持续一段轮次。在 900 轮次左右，累计奖励又迅速上升，当上升至一定阶段后开始波动。可以发现在某些轮次附近波动范围较大，呈现迅速下降又迅速上升的趋势。随着训练的进行，在经历过一段时间后，淡蓝色区域逐渐缩小，波动逐渐平稳，累计奖励值收敛在 47000 左右

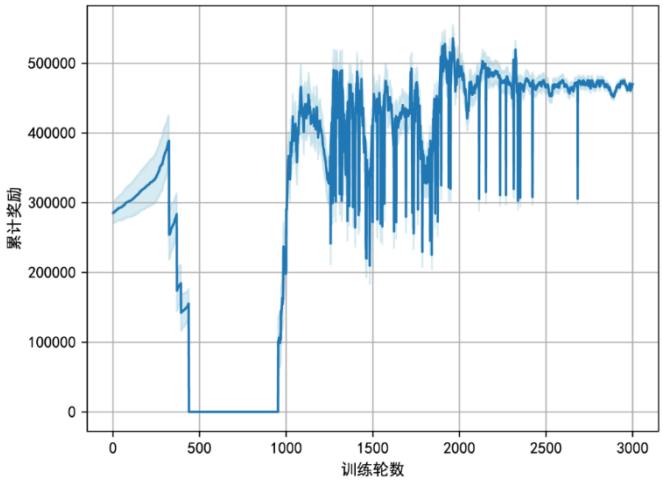


图 5-6 累计奖励随训练过程变化趋势

Figure 5-6 trend of the reward function with the training process

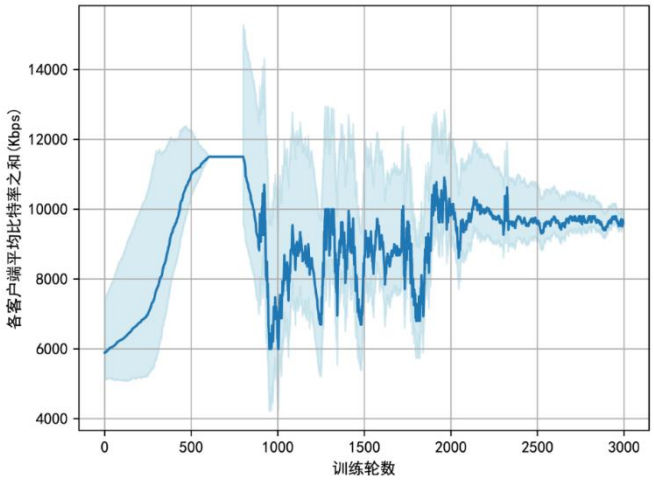


图 5-7 各客户端平均比特率之和随训练过程变化趋势

Figure 5-17 the sum of the average bit rates of each client changes with the training process

如图 5-7 所示，随着训练轮数的增加，各客户端平均比特率之和先平稳上升，上升至最高点后在一定轮次内保持不变。接着骤然下降，下降至最低点后又逐渐上升，在经历一段时间波动后逐渐收敛在 9600Kbps 左右。如图 5-8 所示，在训练轮次到达 300 左右后，开始出现卡顿，并随着训练的进行，总卡顿时间迅速上升，当上升至最高点后，在一定时间内保持稳定。在训练轮次到达 800 左右时，总卡顿时间迅速下降，直到为 0。在训练 1000 轮次后，总卡顿时间在经历过一段时间较小范围的波动后，逐渐稳定为 0。如图 5-9 所示，各客户端平均比特率标准差在训练的初始阶段呈现剧烈波动形式。在训练 500 轮次左后，下降至最低点，并保持不变。在 800 轮次左后又徐速上升，在经历过一段时间的波动后，逐渐稳定在较低的范围，提升总奖励。

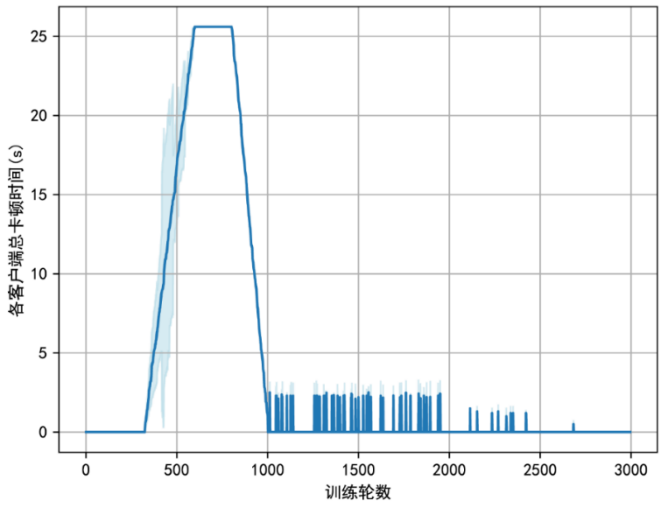


图 5-8 总卡顿时间随训练过程变化趋势

Figure 5-8 the total lag time varies with the training process

结合图 5-6，图 5-7，图 5-8 和图 5-9 可以发现在训练的初始阶段，智能体采取连续提高比特率和公平分配吞吐量的策略来获得更高的累计奖励。此时各客户端平均比特率之和不算上升，各客户端平均比特率标准差不断降低。但随着训练的进行，累计奖励不但没有上升，反而迅速降低，这和算法设计的初衷相符合。分析原因在于，当各客户端平均比特率上升到一定阶段后，客户端分配的吞吐量已经无法满足下载要求，导致客户端开始出现卡顿。而卡顿时不被允许的，此时该客户端的 QoE 为 0，因此总奖励会迅速下降。在经历过一段时间后，智能体逐渐意识到，仅仅以提升比特率和减少各客户端的平均比特率标准

差为目的，并不能实现更高的奖励。此时智能体开始放弃原有的策略，转而将目标放在减少卡顿上。随后各客户端平均比特率开始下降，总卡顿时间开始下降，累计奖励开始上升。通过不断训练，智能体最终学会如何保证公平性和不卡顿的前提下，实现最大化累计奖励，即最大化总 QoE。

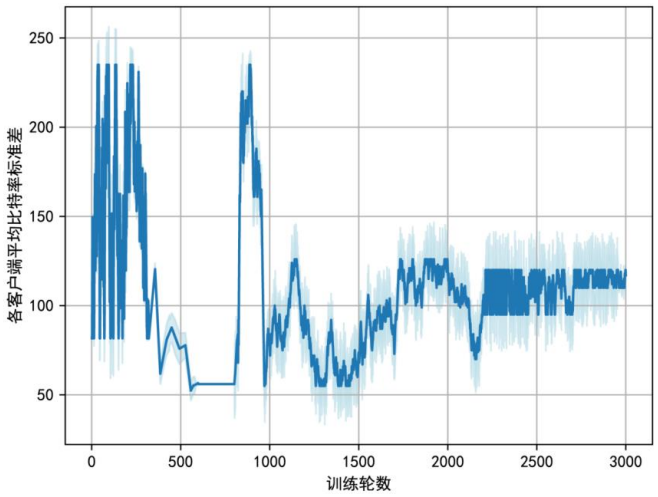


图 5-9 各客户端比特率标准差随训练过程变化趋势

Figure 5-9 The standard deviation of the bitrate of each client changes with the training process

5.4.5 算法性能对比

图 5-10 到图 5-15 显示了，本文算法与对比算法在总 QoE、QoE 标准差、各客户端平均比特率之和、总卡顿时间以及总比特率切换方面的性能对比。

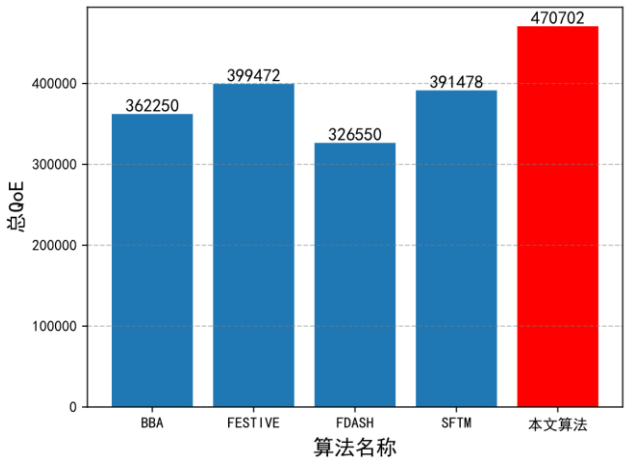


图 5-10 总 QoE

Figure 5-10 Total QoE

如图 5-10 所示，本文算法在所有客户端的总 QoE 比较中明显高于对比算法，相比于 BBA、FESTIVE、FDASH 和 SFTW 算法，总 QoE 分别提升了 29.94%rutu1、17.83%、44.14%和 20.24%。这表明本文算法具有优异的 QoE 性能，在播放视频时，能够提供更多的体验质量。

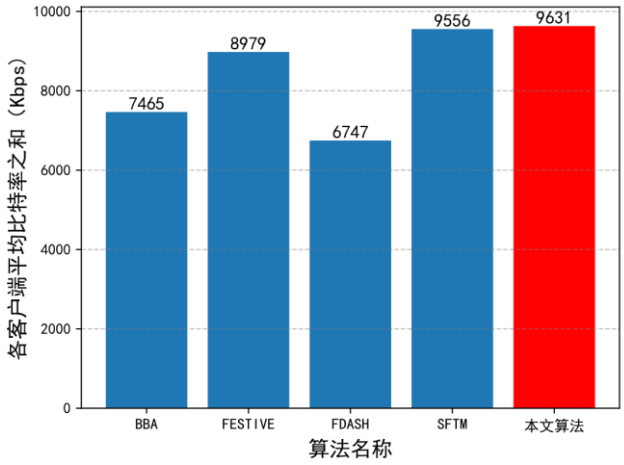


图 5-11 各客户端平均比特率之和

Figure 5-11 The sum of the average bit rates of each client

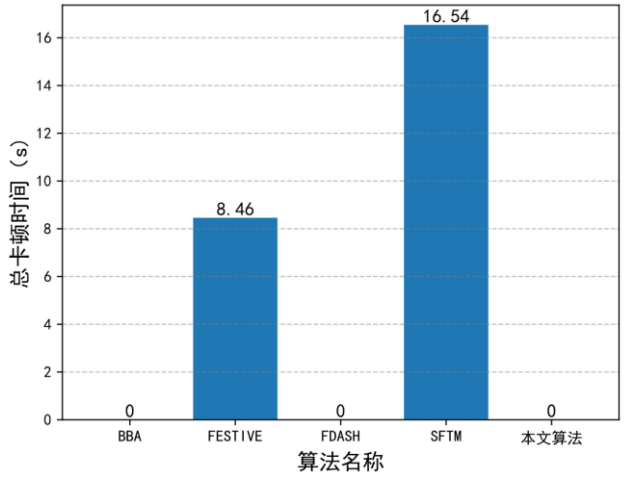


图 5-12 总卡顿时间

Figure 5-12 Total rebuffering time

如图 5-11 所示，本文算法在各客户端平均比特率之和指标对比中，明显优于 BBA、FESTIVE 和 FDASH 算法，分别提升了 29.02%、7.26%和 42.74%。尽管相比于 SFTM 算法，该项指标没有明显提升，但结合图 5-12 和图 5-13 发现，SFTM 算法在所有客户端的总卡顿时间上最长，达到了 16.54s，同时具有最大的

各客户端比特率切换之和。这表明 SFTM 算法在播放视频过程中极易发生卡顿，并且视频平滑度相比于其他算法较低。而相比于 SFTM 算法，本文算法的总卡顿时间为 0，且具有相对较低的各客户端比特率切换之和。因此，如图 5-10 中所示，SFTM 算法在总 QoE 上没有明显的优势，仅仅高于 BBA 和 FDASH。

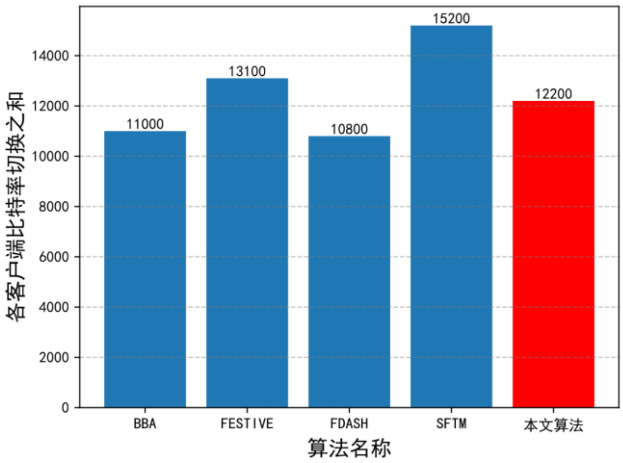


图 5-13 各客户端比特率切换之和

Figure 5-13 Sum of bitrate switching for each client

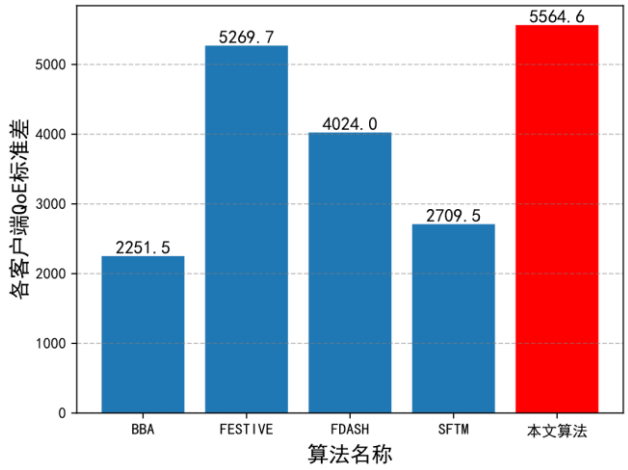


图 2-14 各客户端 QoE 标准差

Figure 2-14 Standard deviation of QoE for each client

如图 5-14 和 5-15 所示，本文算法在各客户端 QoE 标准差和各客户端平均比特率标准差指标方面明显高于 BBA，FDASH 和 SFTM。结合图 5-10 到图 5-14 分析发现 BBA 和 FDASH 算法中各客户端更倾向于选择较低的比特率，因此各客户端平均比特率较为均匀，比特率切换较低，各客户端 QoE 差异较小。而 SFTM 算法中各客户端更倾向于选择较高的比特率，虽然客户端平均比特率标

准差较低，但导致了较长的卡顿时间，从而导致 QoE 较低，因此各客户端 QoE 标准差也较低。

综上所述，在保证所有客户端不卡顿和一定公平性的条件下，本文算法能够最大化所有用户的 QoE，因此验证了本文算法的优异性能。

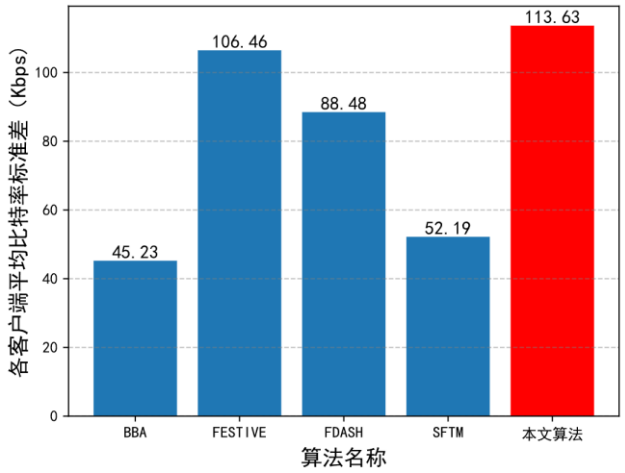


图 5-15 各客户端平均比特率标准差

Figure 5-15 Standard deviation of the average bit rate of each client

5.5 本章小结

本章首先描述在多客户端竞争同一网络带宽资源场景中现有 ABR 算法的不足以及本章的研究目的。其次介绍了多智能体深度强化学习，其中包括强化学习理论、多智能体强化学习以及多智能体强化学习框架。针对现有的 ABR 算法在多客户端竞争同一网络带宽资源时，带宽分配不均匀且用户 QoE 较低的问题，提出了一种基于 COMA 的多客户端流媒体 QoE 优化算法。该算法能够有效提升客户端在无线环境下的 QoE 且保证了用户之间带宽分配的公平性，同时避免了所有客户端卡顿发生的情况。在算法设计中首先对问题进行了 POMDP 建模，然后描述了本章的算法，其中包括对 COMA 算法的介绍以及对任务的状态空间，动作空间和全局奖励函数的设计，最后通过 COMA 算法对多智能强化学习问题进行求解。为了验证算法的有效性，在仿真实验平台下进行实验，并与现有的 ABR 算法进行多种性能指标的对比。实验结果表明本章所提算法在多个指标方面都具有较好的性能。

第 6 章 总结与展望

6.1 工作总结

无线网络技术的发展为移动流媒体技术奠定了重要基础，然而也为 ABR 算法的研究带来了严峻的挑战。本文首先对流媒体传输技术的发展进行了简单介绍，并描述 MPEG-DASH 技术的主要内容以及现有 ABR 算法的研究现状。然后针对 ABR 算法在无线环境下面临的挑战，提出了相应的解决方案，以改善无线环境下的用户观看体验质量。主要工作总结如下：

1.针对现有的吞吐量预测器在无线环境下无法进行准确的吞吐量预测问题，设计并实现了一种基于 Informer 的吞吐量预测器 TPMI。与传统的预测方法相比，考虑了更多的影响吞吐量预测的因素，包括网络连接类型、信号轻度、视频切片大小、历史吞吐量和播放器状态。实验结果表明，与基于 HM、LSTM 和 MLR 的预测器相比，TPMI 具有更好的预测性能，预测准确率得到明显提升。将 TPMI 集成到 MPC 算法中，与 Pensieve 和 BBA 算法相比，用户的 QoE 得到有效提升。

2.针对在无线网络带宽剧烈且频繁波动时，现有的 ABR 算法选择的视频平均比特率低且比特率切换频繁，从而导致较低的视频质量和平滑度，提出了一种基于 QoE 感知的替换补偿算法。算法的核心思想是在带宽改善时,以更高比特率的视频切片动态替换缓冲区中尚未播放的低比特率视频切片。仿真实验表明，融合了该算法的 BOLA-E 算法和 THROUGHPUT 算法性能得到有效提升。

3.针对现有的 ABR 算法在多客户端竞争同一网络带宽资源时，带宽分配不均匀且用户 QoE 较低的问题。提出了一种基于 COMA 的多客户端流媒体 QoE 优化算法。该算法基于集中式训练和分布式执行(CTDE)的多智能体强化学习框架,对多客户端比特率决策问题进行分布式部分可观测马尔可夫模型(Decentralized Partially Observable Markov Decision Process ,Dec-POMDP)建模，并通过基于反事实多智能体 COMA 策略梯度算法进行求解。仿真实验表明，与 FESTIVE、SFTM 等算法相比，视频卡顿时间明显减少，并且在维护各用户之间公平性的基础上有效提升了用户的 QoE。

6.2 工作展望

本文对现有 ABR 算法进行了研究，针对无线网络环境下带宽变化的随机性、时变性等特点，提出了一种适用于无线环境下网络吞吐量的预测器，并且基于现有自适应算法提出了一种基于 QoE 感知的替换补偿算法，同时针对多客户端竞争同一网络带宽资源的场景，提出了一种基于深度多智能体深度强化学习的 QoE 优化算法。在仿真实验下，本文所设计的吞吐量预测器具有准确的预测性能，所提算法都能够在实现提升无线环境下用户 QoE 的目的。为了进一步提升用户在无线环境下的 QoE,未来的工作主要集中在以下几点：

1.在多客户端场景中，将本文所提出的吞吐量预测器以及基于 QoE 感知的替换补偿算法融合到基于深度多智能体深度强化学习的 QoE 优化算法中，实现更好的性能。

2.虽然本文设计的预测器以及提出基于 QoE 感知的替换补偿算法和基于深度多智能体深度强化学习的 QoE 优化算法在仿真环境下具有较好的性能，但没有实际部署在 DASH 中，因此如何将本文所设计的吞吐量预测器以及所以出的算法部署在 DASH 中将是未来的工作之一。

参考文献

[1] Cisco Visual Networking Index: Forecast and Trends, 2017-2022 Whitepaper https://www.cisco.com/c/en/us/solutions/collateral/service-provider.

[2] Hung, Nguyen Viet et al. Flexible HTTP-based Video Adaptive Streaming for good QoE during sudden bandwidth drops[C]. EAI Endorsed Trans. Ind. Networks Intell. Syst，2023，10(2): e3.

[3] LEBRETON P, YAMAGISHI K. Quitting ratio-based bitrate ladder selection mechanism for adaptive bitrate video streaming[J]. IEEE Transactions on Multimedia, 2023, 25: 8418-8431.

[4] Nguyen, Minh et al. H2BR: an HTTP/2-based retransmission technique to improve the QoE of adaptive video streaming[C]. Proceedings of the 25th ACM Workshop on Packet Video, 2020: 1–7.

[5] Tashtarian, Farzad et al. LALISA: Adaptive Bitrate Ladder Optimization in HTTP-based Adaptive Live Streaming[C]. NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium , 2023: 1-9.

[6] Khan, Koffka and Wayne Goodridge. Comparative study of One-Shot Learning in Dynamic Adaptive Streaming over HTTP : A Taxonomy-Based Analysis[J]. International Journal of Advanced Networking and Applications, 2023, 15(01): 5822-5830.

[7] 陈自强.基于自适应流媒体传输的直播系统建设[J].信息技术与信息化，2024(02):147-150.

[8] You, Dongho et al. MPEG-DASH MPD for Tile-based Hybrid Stereoscopic 360-degree Video Streaming[C]. 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN), 2018: 682-684.

[9] Bentaleb, Abdelhak et al. A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP. IEEE Communications Surveys & Tutorials, 2019, 21(01): 562-585.

[10] Kua J, Armitage G, Branch P. A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP[J]. IEEE Communications Surveys & Tutorials, 2017, 19(3): 1842-1866.

[11] Tadahal S S, Gummadi S V, Prajapat K, et al. A Survey On Adaptive Bitrate Algorithms and Their Improvisations[C]. 2021 International Conference on Intelligent Technologies (CONIT), 2021: 1-7.

[12] Seufert M, Egger S, Slanina M, et al. A survey on quality of experience of HTTP adaptive streaming[J]. IEEE Communications Surveys & Tutorials, 2014, 17(1): 469-492.

[13] Chenghao Liu, Bouazizi, et al. Rate adaptation for adaptive HTTP streaming[C]. ACM SIGMM Conference on Multimedia Systems, 2011: 169-174.

[14] Chenghao Liu, Bouazizi, et al. Rate adaptation for dynamic adaptive streaming over HTTP in content distribution network[J]. Signal Process. Image Commun, 2012, 27 (4): 288-311.

[15] Benjamin Rainer，Stefan Lederer，et al. A seamless Web integration of adaptive HTTP streaming[C]. 2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO), 2012: 1519-1523.
[16] Li Z, Zhu X, Gahm J, et al. Probe and adapt: Rate adaptation for HTTP video streaming at scale[J]. IEEE Journal on Selected Areas in Communications, 2014, 32(4): 719-733.
[17] Jiang J, Sekar V, Zhang H. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive[C]. Proceedings of the 8th international conference on Emerging networking experiments and technologies, 2012: 97-108.
[18] Xiufeng Xie, Xinyu Zhang，et al. piStream: Physical Layer Informed Adaptive Video Streaming over LTE[C]. Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, 2015: 413-425.
[19] Konstantin Miller, Abdel-Karim Al-Tamimi, et al. QoE-Based Low-Delay Live Streaming Using Throughput Predictions[J]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2016,13(1): 1-4.
[20] Christian Sieber，Tobias Hossfeld，Thomas Zinner，et al. Implementation and user-centric comparison of a novel adaptation logic for DASH with SVC[C]. 2013 IFIP/IEEE International Symposium on Integrated Network Management,2013: 1318-1323.
[21] Huang T Y, Johari R, McKeown N, et al. A buffer-based approach to rate adaptation: Evidence from a large video streaming service[C]. Proceedings of the 2014 ACM conference on SIGCOMM, 2014: 187-198.
[22] Christopher Müller, Stefan Lederer,et al. Oscillation compensating dynamic adaptive streaming over HTTP[C]. 2015 IEEE International Conference on Multimedia and Expo (ICME), 2015: 1-6.
[23] Kevin Spiteri, Rahul Urgaonkar,et al. BOLA: Near-Optimal Bitrate Adaptation for Online Videos[C]. The 35th Annual IEEE International Conference on Computer Communications, San Francisco, 2016: 1-9
[24] Yadav, Praveen Kumar, et al. QUETRA: A Queuing Theory Approach to DASH Rate Adaptation[C]. Proceedings of the 25th ACM international conference on Multimedia, 2017: 1130–1138.
[25] K Miller, E Quacchio, et al. Adaptation algorithm for adaptive streaming over HTTP[C]. 2012 19th International Packet Video Workshop (PV), 2012: 173-178.
[26] J Doyle, B Francis,et al. Feedback Control Theory［M］. New York, NY, USA: Macmillan, 2013.
[27] Xiaoqi Yin, Abhishek Jindal,et al. A control theoretic approach for dynamic adaptive video streaming over HTTP[C]. 2015 ACM Conference on Special Interest Group on Data Communication, 2015: 325–338.
[28] A. Beben, Piotr Wisniewski,et al. ABMA+: Lightweight and efficient algorithm for HTTP adaptive streaming[C]. Proceedings of the 7th International Conference on Multimedia Systems, 2016: 1-11.
[29] Cong Wang, Amr Rizk, Michael Zink.SQUAD: A spectrum-based quality adaptation for dynamic adaptive streaming over HTTP[C]. Proceedings of the 7th

International Conference on Multimedia Systems, 2016: 1-12.
[30] Sobhani, Ashkan, Abdulsalam Yassine,et al. A video bitrate adaptation and prediction mechanism for HTTP adaptive streaming[C]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM),2017,13(2): 1-25.
[31] Bentaleb A, Begen AC,et al. Want to play DASH? A game theoretic approach for adaptive streaming over HTTP[C]. Proceedings of the 9th ACM Multimedia Systems Conference, 2018: 13-26.
[32] Min Xing, Siyuan Xiang ,et al. A real-time adaptive algorithm for video streaming over multiple wireless access networks[J]. IEEE Journal on Selected Areas in Communications, 2014, 32(4): 795–805.
[33] Ayub Bokani, Mahbub Hassan,et al. Optimizing HTTPbased adaptive streaming in vehicular environment using Markov decision process[J]. IEEE Transactions on Multimedia, 2015 ,17(12):2297–2309.
[34] Chao Zhou, Chia-Wen Lin,et al. mDASH: A Markov decision-based rate adaptation approach for dynamic HTTP streaming[J]. IEEE Transactions on Multimedia , 2016, 18(4): 738–751.
[35] Hongzi Mao, R Netravali, et al. Neural Adaptive Video Streaming with Pensieve[C]. Proceedings of the Conference of the ACM Special Interest Group on Data Communication, 2017: 197–210.
[36] Gadaleta M, Chiariotti F, Rossi M, et al. D-DASH: A deep Q-learning framework for DASH video streaming[J]. IEEE Transactions on Cognitive Communications and Networking, 2017, 3(4): 703-718.
[37] Sengupta S, Ganguly N, Chakraborty S, et al. HotDASH: Hotspot aware adaptive video streaming using deep reinforcement learning[C]. 2018 IEEE 26th International Conference on Network Protocols (ICNP). IEEE, 2018: 165-175.
[38] Huang T, Zhou C, Zhang R X, et al. Comyco: Quality-aware adaptive video streaming via imitation learning[C]. Proceedings of the 27th ACM international conference on multimedia, 2019: 429-437.
[39] Guo Y, Yu F R, An J, et al. Adaptive bitrate streaming in wireless networks with transcoding at network edge using deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2020, 69(4): 3879-3892.
[40] Gatimu K, Lee B. qMDP: DASH Adaptation using Queueing Theory within a Markov Decision Process[C]. 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC), 2021: 1-6.
[41] LI W, HUANG J, LYU W, et al. RAV: Learning-Based Adaptive Streaming to Coordinate the Audio and Video Bitrate Selections[J]. IEEE Transactions on Multimedia, 2023, 25: 5662-5675.
[42] LI W, LI X, XU Y, et al. MetaABR: A Meta-Learning Approach on Adaptative Bitrate Selection for Video Streaming[J]. IEEE Transactions on Mobile Computing, 2024, 23(3): 2422-2437.
[43] Iren, Ecem and Aylin Kantarci. Content Aware Video Streaming with MPEG DASH Technology[J]. TEM Journal, 2022, 11(2): 611-619.
[44] Krum, Andrew et al. FAST DASH: Program overview and key findings from initial

technology evaluations[J]. Accident analysis and prevention, 2019, 124: 113-119 .
[45] Pecioski, Damjan et al. An overview of reinforcement learning techniques[C]. 2023 12th Mediterranean Conference on Embedded Computing (MECO), 2023: 1-4.
[46] W. Li, X. Li, C. Chen and A. Song. Overview of Reinforcement Learning for Person Re-Identification[J]. in IEEE Transactions on Biometrics, Behavior, and Identity Science, 2023, 5(1): 105-114.
[47] 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.
[48] Sikora, Janusz and Renata Wagnerova. Overview of Reinforcement Learning and its Application in Control Theory[C]. 2020 21th International Carpathian Control Conference (ICCC), 2020: 1-4.
[49] 邹启杰,蒋亚军,高兵,等.协作多智能体深度强化学习研究综述[J].航空兵器,2022,29(06):78-88.
[50] Liang, Le et al. Spectrum Sharing in Vehicular Networks Based on Multi-Agent Reinforcement Learning[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(10): 2282-2292.
[51] Canese, Lorenzo et al. Multi-Agent Reinforcement Learning: A Review of Challenges and Applications[J]. Applied Sciences, 2021, 11(11): 4948.
[52] Schmidt, Lukas M. et al. An Introduction to Multi-Agent Reinforcement Learning and Review of its Application to Autonomous Mobility[C]. 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), 2022: 1342-1349.
[53] Xiang, Xuanchen and Simon Foo. Recent Advances in Deep Reinforcement Learning Applications for Solving Partially Observable Markov Decision Processes (POMDP) Problems: Part 1 - Fundamentals and Applications in Games, Robotics and Natural Language Processing[J]. Mach. Learn. Knowl. Extr, 2021, 3(3): 554-581.
[54] 陈人龙,陈嘉礼,李善琦,等.多智能体强化学习方法综述[J].信息对抗技术,2024,3(01):18-32.
[55] Xu, Chi et al. Multi-Agent Reinforcement Learning With Distributed Targeted Multi-Agent Communication[C]. 2023 35th Chinese Control and Decision Conference (CCDC), 2023: 2915-2920.
[56] Foerster, Jakob N. et al. Counterfactual Multi-Agent Policy Gradients[C]. AAAI Conference on Artificial Intelligence, 2017, 363:2974-2982.
[57] Song X. MADDPG: an efficient multi-agent reinforcement learning algorithm[C]. International Conference on Advanced Algorithms and Neural Networks, 2022, 12285: 50-55.
[58] Ma, Yanhao and Jie Luo. Value-Decomposition Multi-Agent Proximal Policy Optimization[C]. 2022 China Automation Congress (CAC), 2022: 3460-3464.
[59] Sunehag, Peter et al. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward[C]. Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, 2018: 2085–2087.
[60] Rashid T, Samvelyan M, De Witt C S, et al. Monotonic value function factorisation for deep multi-agent reinforcement learning[J]. Journal of Machine Learning

Research, 2020, 21(178): 1-51.
[61] Zhou H, Zhang S, Peng J, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting[C]//Proceedings of the AAAI conference on artificial intelligence, 2021, 35(12): 11106-11115.
[62] X. K. Zou, et al. Can Accurate Predictions Improve Video Streaming in Cellular Networks[C]. Workshop on Mobile Computing Systems and Applications, 2015: 57-62.
[63] J. Schmid, et al. A Deep Learning Approach for Location Independent Throughput Prediction[C]. 2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE), 2019: 1-5.
[64] Liu, Y. et al. Throughput Prediction-Enhanced RL for Low-Delay Video Application[C]. 2022 18th International Conference on Mobility, Sensing and Networking (MSN) , 2022: 728-735.
[65] Vaswani, Ashish et al. Attention is All you Need[C]. Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017: 6000–6010.
[66] G. Lv, et al. Lumos: towards Better Video Streaming QoE through Accurate Throughput Prediction[C]. IEEE INFOCOM 2022-IEEE Conference on Computer Communications, 2022: 650-659.
[67] F. Y. Yan, et al. Learning in situ: a randomized experiment in video streaming[C]. Symposium on Networked Systems Design and Implementation, 2020: 495–512.
[68] Gerui Lv,Qinghua Wu, et al. Lumos: towards Better Video Streaming QoE through Accurate Throughput Prediction[C]. IEEE INFOCOM 2022-IEEE Conference on Computer Communications, 2022: 650-659.
[69] Spiteri K, Sitaraman R, Sparacio D. From theory to practice: Improving bitrate adaptation in the DASH reference player[J]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2019, 15(2): 1-29.
[70] Riiser, Håkon et al. Commute path bandwidth traces from 3G networks: analysis and applications. Proceedings of the 4th ACM Multimedia Systems Conference,2013: 114-118.
[71] The ns-3 development team, “ns-3 network simulator.” https://www.nsnam.org/docs/release/3.32/tutorial/html/ introduction.html. Accessed: 2020-11-28.
[72] Liu C, Bouazizi I, Hannuksela M M, et al. Rate adaptation for dynamic adaptive streaming over http in content distribution network[J]. Signal Processing: Image Communication, 2012, 27(4): 288-311.
[73] Vergados D J, Michalas A, Sgora A, et al. A fuzzy controller for rate adaptation in mpeg-dash clients[C]. 2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC), 2014: 2008-2012.

攻读学位期间参与的科研项目

[1] “飞行自组织网络服务质量跨层优化机制研究”，安徽省教育厅自然科学研究重点项目，项目编号：No.KJ2020A0361。

[2] “基于 5G 网络的智能交互展示系统研究”产学研项目，项目编号：HX-2023-11-088。

攻读学位期间发表的学术论文目录

[1] Jiang Liu, Jun Li, et al. TPMI：Accurate Throughput Prediction for Better bitrate selection in Adaptive Video Streaming [C]. The 2nd International Conference on Sensing, Measurement, Communication and Internet of Things Technologies (SMC-IoT) ,2023(第一作者, EI 会议，已发表)

[2] Jiang Liu, Jun Li, et al. QoE-aware Adaptive video Streaming by Replacement Compensation Algorithm [J]. IEICE Transactions on Communications,2023(第一作者, SCI 源刊，在投)

致谢

三年的研究生旅程已然接近尾声,虽然时光短暂,但这段日子却丰富多彩。回首这段历程,充满了憧憬、奋斗、坚韧不拔以及克服困难的时刻,汗水与泪水交织,辛勤的耕耘终将带来丰硕的成果。即将毕业之际,我想向培育我成长的母校表达由衷的感谢,同时也向所有陪伴我走过这段路、关心我、支持我的人们致以最真挚的谢意和祝福。

首先，我尤为的感激我的导师，李钧副教授。感谢您在这三年中对我的学术研究和生活中的指导以及帮助。遥想当初刚进学校的时候，我还在为没有导师而苦恼，偶然听身边的人说，李钧老师为人谦和友善，于是我便与您联系，很荣幸我成功成为你们学生。随后很快的时间里，我们便确定了研究方向，从一开始教我怎样阅读论文到实现小论文的创新，您总能耐心的对我进行指导，总能及时的对我进行批评指正。在研一学年里，您督促我完成了对机器学习，深度学习以及强化学习内容的学习，在这过程中，我很多次没有完成你布置的任务，虽然您当时很生气，但后面总能耐心指导我，后来这些知识我的学术研究和项目工作中发挥了重要的作用在研二的学年里，您指导我完成了第一篇小论文的写作。在这期间，您指导我对小论文进行过数十次的修改完善，让我学会了很多，为我在后面的学术写作中打下了坚实的基础。除此之外，作为一名专硕学生，您带领我参加公司的项目，不仅锻炼了我的动手能力，同时让我明白自身的不足。在研三的学年里，您耐心的指导我硕士论文的创新以及写作，每一次的修改完善中您总能详细的批注，从而帮助我成功完成大论文的写作。在生活中，您也总能给予我帮助。尤其是当我因患新冠在校医院隔离的这段时间里，您不仅给我送了水果，并且每天都询问我的病情，在这里，我再次感谢您对我生活中关心和帮助。

其次，我想要向 Prise 小组的各位老师和同学们表示感谢。每周六组会的论文分享和进度汇报，更加督促我学习和帮助我进步。非常感谢 Prise 小组中王老师、卢老师、唐老师以及其他老师在组会上给予的学术指导以及其他帮助。同时也很感谢课题组中的同学们，感谢你们共同营造了一种和谐、友好的学习氛围，同时感谢你们的知识分享，真的让我收获很多。

再次，我要感谢我的师弟师妹们，真的很想说遇到你们真的很好，是你们的陪伴，让我的研究生生活不在单调枯燥。感谢你们的陪伴和付出，感谢在我遇到困难时，你们总能给予我鼓励和支持。同时，也很感谢你们在学术和生活中的交流和分享，三年的时间真的很快，甚至有点不舍，在这里祝福你们在科研的道路上一帆风顺，成果频出。

此外，我想要特别感谢我的父母和女朋友。每当我遇到困难或情绪低落时，你们总是会耐心地听我诉说，给我鼓励和建议，这对我来说真的太重要了，是你们让我觉得不是一个人在前行；当我取得成就时，你们会真心地为我感到高兴和安慰，并激励我保持初心，继续前进。

最后，我想向审阅本论文的专家和答辩的老师们表示感谢。我非常感谢您们在忙碌的工作中抽出宝贵的时间，对我的论文进行批评和指导。