TAROT：面向视频流媒体的优化驱动自适应 FEC 参数调优

Jashanjot Singh Sidhu*康考迪亚大学，加拿大
jashanjot.sidhu@mail.concordia.ca

Aman Sahu*康考迪亚大学，加拿大
aman.sahu@mail.concordia.ca

Abdelhak Bentaleb 康考迪亚大学，加拿大
abdelhak.bentaleb@concordia.ca

注意： 本文是我们已被 **接收的论文的扩展版本**，该论文发表于**第 17 届 ACM多媒体系统会议（ACM MMSys 2026）**。

摘要

前向纠错（FEC）对于保护视频流媒体免受数据包丢失影响仍然至关重要，但大多数实际部署仍依赖静态、粗粒度的配置，无法应对丢包率、有效吞吐量或客户端缓冲区水位的快速变化。这些僵化的设置往往会造成低效：稳定时段中不必要的冗余会抑制吞吐量，而突发丢包期间保护又不足；尤其是浅缓冲区和过大的数据块会增加卡顿风险。为应对这些挑战，我们提出了TAROT，这是一种跨层、由优化驱动的前向纠错控制器，可按分段选择冗余、块大小和符号化方式。TAROT与编解码器无关，支持里德–所罗门、RaptorQ 和基于异或的编码，并使用细粒度评分模型评估预先计算的候选集。该评分函数综合考虑传输层丢包和有效吞吐量、应用层缓冲区动态以及数据块级时序约束，从而对覆盖不足、开销过高和数据块完成速度过慢进行惩罚。为实现逼真的测试，我们扩展了 SABRE 模拟器 [1]，新增两个模块：可重放多种丢包模式的高保真数据包丢失生成器，以及支持任意编码/参数组合的模块化前向纠错基准测试层。在低延迟直播（LLL）和视频点播（VoD）流媒体模式、多种网络轨迹以及多种自适应码率算法下，TAROT 最多可将前向纠错开销降低 43%，同时将感知质量提高 10 个 VMAF单位，且几乎不增加重新缓冲；与静态前向纠错相比，实现了更优的开销与质量平衡。

CCS 概念

• **信息系统** →**多媒体流**。

关键词

前向纠错、体验质量、跨层、视频点播、LLL。

*前两位作者对本研究的贡献相同。[1]TAROT 代码可在以下地址获取：https://github.com/IN2GM-Lab/TAROT-FEC

允许免费为个人或课堂使用而复制本作品的全部或部分内容（无论数字版还是纸质版），前提是不得为营利或商业利益而复制或分发，且副本须附有本声明及首页的完整引文。对于本作品中由作者以外人士拥有版权的组成部分，必须尊重其版权。注明出处后可摘录。若要以其他方式复制、再版、发布到服务器或分发至列表，则须事先获得具体许可和/或支付费用。请通过permissions@acm.org 申请许可。*MMSys '26，2026*年*4*月*4*日至*4*月*8*日，中国香港。© 2026 版权所有归所有者/作者所有。出版权已授权给 ACM。ACM ISBN 978-1-4503-XXXX-X/18/06https://doi.org/XXXXXXX.XXXXXXX

ACM参考格式：
Jashanjot Singh Sidhu、Aman Sahu 和 Abdelhak Bentaleb。2026 年。TAROT：面向视频流媒体的优化驱动型自适应 FEC 参数调优。发表于*ACM* 多媒体系统会议 *2026*（*MMSys '26*），*2026*年*4*月*4*日至*4*月*8*日，中国香港。ACM，纽约，纽约州，美国，15 页。https://doi.org/XXXXXXX.XXXXXXX

1 引言

视频流量的快速增长 [46]给内容提供商带来了前所未有的压力，要求其持续提供高体验质量（QoE），尤其是在 4K 和高帧率格式日益普及的情况下。通过当今异构接入网络（尤其是 LTE 和 5G）传输此类内容仍面临挑战。尽管这些网络具有较高的峰值吞吐量，却会出现剧烈的短期波动、偶发拥塞和不可忽视的丢包 [3, 13, 28, 54], 这些因素都会降低视频播放器可用的有效吞吐量。因此，在此类波动下维持稳定的高质量播放变得愈发困难。

历来，大多数体验质量优化工作都聚焦于流媒体协议栈的两个层面。在客户端侧，HTTP自适应流媒体 [7]（HAS）算法会根据观测到的带宽和缓冲区状态，选择“最佳”比特率表示版本。在传输侧，拥塞控制 [29] 机制会调整流量，以避免队列积压，并根据网络容量调整发送速率。尽管这些机制在带宽波动时有效，但从根本上说并不适用于有损环境。即使是轻微的丢包，也可能引发有效吞吐量超线性崩溃、降低吞吐量估计，并导致自适应码率算法发生振荡或退回到低质量表示版本，而不考虑可用的峰值带宽。在波动剧烈的 LTE 和 5G 环境中，这些挑战会进一步加剧；此时重传会适得其反——增加时延、降低有效吞吐量，并带来拥塞—丢包螺旋的风险。

这种不匹配促使人们采用前向纠错 [18, 44] (FEC)，将其作为视频传输的补充可靠性机制。通过主动注入修复符号，前向纠错使接收端无需依赖传输层重传即可恢复丢失的数据。理论上，这应能减少播放卡顿，增强对突发丢包的韧性，并稳定 HTTP 自适应流媒体客户端中的表示选择。然而，前向纠错在流媒体系统中的实际部署仍受若干长期存在的局限所困扰。大多数现有系统 [23, 49, 59] 依赖静态前向纠错参数——固定的块大小、固定的符号大小以及固定的

冗余率——这些参数在整个流媒体会话期间都保持不变。这类配置易于部署，但在实践中缺乏灵活性。网络状况稳定时，静态前向纠错会浪费带宽，降低自适应码率逻辑可用的有效吞吐量。网络出现丢包时，同样固定的冗余量往往又不足以应对，导致解码失败并触发重传，反而违背了采用前向纠错的初衷。

为克服这些局限，研究者提出了多种基于启发式和学习的冗余控制器。启发式方法 [14] 会根据观测到的丢包或估计的拥塞情况调整冗余比率，但通常将前向纠错视为一个单一的标量“开销旋钮”，无法调整底层数据块结构。近期的强化学习方法 [6, 26]虽然提供了更大的灵活性，但仍受限于固定的编码设计，且未考虑分段级时序约束、缓冲区状态，或丢包导致的有效吞吐量非线性崩溃。因此，现有方案难以在各种网络状况下平衡恢复能力和开销效率，尤其是在当今变化剧烈的5G环境中。

这些挑战揭示了视频流媒体可靠性机制设计中的一个根本性缺口：前向纠错不仅必须调整所添加冗余的数量，还必须决定其组织方式、应用时机，以及它与缓冲区动态、分段时长和丢包导致的有效吞吐量下降之间的相互作用。弥补这一缺口需要一种细粒度的跨层方法，能够基于实时传输和播放信号进行逐分段决策。为解决这些局限，我们提出 TAROT[2]，这是一种专为易变网络上的HTTP自适应流媒体视频流传输设计的细粒度跨层前向纠错控制器。TAROT 不同于传统的冗余控制器：它在分段级别运行，并联合考量丢包、冗余开销、编码延迟和分段时序约束。TAROT 不会将前向纠错视为单一的标量开销旋钮，而是针对每个分段评估多个码族和块配置（n，k，S），其中n 表示源符号的数量，k 表示修复符号的数量，S 表示符号大小，并选择能使多目标成本函数最小化的配置，以平衡开销、恢复能力和感知影响。

为支持对此类逐分段决策进行贴近真实的评估，我们扩展了广泛采用的SABRE [53] 仿真器，并开发了*SABRE–FEC*，这是一个模块化的分组级前向纠错框架，可同时建模冗余导致的带宽膨胀以及生成修复符号的计算开销。SABRE–FEC集成了解析丢包模型，以刻画丢包条件下有效吞吐量的非线性崩塌，并将编码延迟直接纳入分段下载时间。这使得能够在HTTP自适应流逻辑中对细粒度FEC策略进行可复现、可控且与实现无关的评估。本文的主要贡献有以下三个方面：

(1) 我们开发了 TAROT，这是一种逐分段自适应前向纠错控制器，它将前向纠错参数选择建模为一个受约束的优化问题，并通过候选枚举高效求解。TAROT 会为每个分段联合选择前向纠错码族和配置（n，k，S），并通过轻量级多目标评分模型综合考虑开销、编码延迟、恢复能力和分段截止时间，从而能够在变化剧烈的 LTE 和 5G 条件下稳定运行。

（2）我们构建了SABRE-FEC，它是SABRE仿真器的扩展，包含模块化的分组级FEC层和解析丢包模型。SABRE-FEC真实地建模了传输字节膨胀与编码延迟，从而能够对细粒度FEC策略进行可控且可复现的评估。

(3) 我们使用多种真实世界轨迹（LTE、5G），并在 LLL 和 VoD 两种流媒体模式下，对 TAROT增强FEC 进行了全面评估。结果表明，与静态FEC基线相比，TAROT 能持续提升感知质量并降低开销。

本文其余部分安排如下：第2 节介绍相关工作，随后在第 TAROT 3 节介绍其设计。第3节之后，第4 节将重点评估TAROT 增强FEC 的性能。第5 节讨论其局限性，第6 节总结全文。

2 相关工作

基于启发式的FEC。Reed-Solomon [23], RaptorQ [1, 59], 和基于异或的擦除码 [49] 等传统前向纠错机制已广泛应用于多媒体系统。基于 RTP/RTCP 的前向纠错 [47], , 包括 RFC 5109 [27] 中标准化的奇偶校验方案，也采用基于规则的冗余分配方式。

基于学习的 FEC。近期研究通过网络遥测或强化学习实现冗余自适应。RL-FEC [61] 根据测得的丢包或延迟调整冗余，而 RL-AFEC [9]则利用强化学习和基于视频多方法评估融合的反馈，学习帧级冗余策略。DeepRS [11] 和 LightFEC [21]利用基于长短期记忆网络的丢包预测，动态调整 RS 冗余；针对非地面网络的 RL-FEC [61] 则学习如何在延迟反馈条件下注入修复分组。P-FEC [60] 结合代内与代间编码，以便在丢包预测不确定时满足服务等级协议的解码保证。R-FEC [26]利用强化学习联合优化 WebRTC 的视频比特率和前向纠错比特率，而基于机器学习的 WebRTC 前向纠错 [17]则采用监督学习，提高移动设备上的能效自适应能力。Tooth [6] 面向云游戏，采用逐帧预测器，根据帧长度和丢包聚合情况分配冗余。尽管这些方法实现了自适应，但它们都基于固定的编码结构运行，且只优化单一冗余参数。它们都没有联合考虑前向纠错族、数据块参数（n、k、S）、编码延迟和分段时序约束，而且所有方法都是针对 RTC 或云游戏工作负载开发的，而非 HTTP自适应流媒体。

用于HTTP自适应流媒体的SABRE仿真器。SABRE [53] 被广泛用于研究 HTTP自适应流媒体中的码率自适应。它对吞吐量、往返时间和缓冲区演化进行建模，但原生不支持数据包级丢包、冗余开销或前向纠错处理时延。

以往的前向纠错系统要么依赖静态参数，要么仅调整冗余比例，而现有模拟器缺乏数据包级前向纠错建模，无法捕捉开销、编码延迟与 HTTP自适应流媒体缓冲区动态之间的相互作用。TAROT 通过逐片段联合优化前向纠错族和块配置（n，k，S），将冗余开销、编码成本、丢失导致的有效吞吐量崩溃以及片段时序约束纳入统一的决策流程，从而解决了这些局限。这使 TAROT 能够提供细粒度、自适应的前向纠错，以适应

[2]TAROT是该设计如下全称的缩写：**T**面向优化驱动的**A**自适应前向纠错参数**R**调优，用于视频**O**流**T**媒体

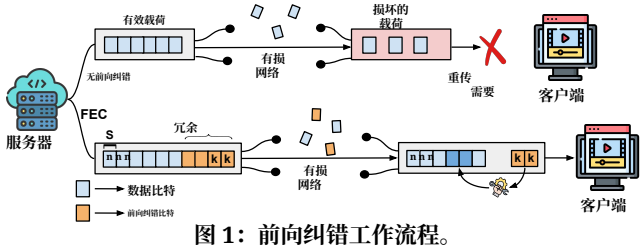

图 1：前向纠错工作流程。

现代 LTE 和 5G 网络的快速变化特性——这是此前所有基于启发式和学习的方法都不具备的能力。

3 TAROT方案

3.1 前向纠错概述

前向纠错通过在原始数据旁传输少量冗余信息，提供了一种主动的丢失恢复机制。这些冗余信息被编码为修复符号，使接收方能够重建数据包或分段中缺失的部分，而无需重传。通过以适度的开销换取及时恢复，前向纠错可减轻丢包的影响，并降低有损网络环境下基于重传的恢复通常带来的时延惩罚。这一机制如图所示1。如图所示，当服务器通过有损信道传输数据时，即使少量符号丢失，也可能导致接收方无法使用整个有效载荷，进而触发重传。由于损坏的载荷无法解码或部分恢复，客户端必须等待完整数据包重新发送，从而产生额外延迟。相比之下，应用前向纠错后，随原始数据一同传输的修复符号使接收方能够恢复数据包中缺失的部分。只要收到足够组合的源符号和修复符号，客户端就能完整重建并解码有效载荷，而无需重传。

无论采用何种编码技术，前向纠错的基本工作原理都不变。设原始有效载荷大小为 P。前向纠错编码器选择符号大小 S，并将有效载荷划分为 n 个源符号，每个符号大小为 S，因此源块占用 nS 字节。随后，编码器应用冗余因子 r，以确定需要生成的修复符号数量 $k = \lceil rn \rceil$。虽然不同的前向纠错方案可能采用不同算法来生成这些修复符号，但它们的结构都相同：在包含 n 个源符号的数据块中增加 k 个修复符号，从而无需重传即可恢复丢失的数据。

由此产生的前向纠错开销直接取决于生成的修复符号数量。由于每个修复符号的大小 S 与源符号相同，编码器增加的冗余数据总量为 kS 字节。鉴于原始有效载荷占用 $P = nS$ 字节，因此前向纠错引入的开销比例为

$$\text{Overhead} = \frac{kS}{nS} = \frac{k}{n} = \frac{\lceil rn \rceil}{n}.$$

这一表达式体现了前向纠错的基本权衡：增加冗余可以更好地防止丢包，但也会带来成比例增加的开销。

在实践中，该开销项 $\frac{k}{n}$ 直接与链路有效吞吐量、缓冲区状态和播放速率相互作用。因此，选择适当的冗余级别 r 成为一个多维优化问题，必须在恢复能力与时延之间取得平衡

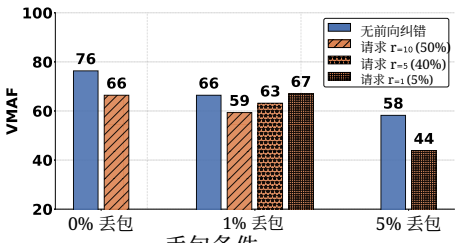

图 2：静态前向纠错参数对客户端体验质量的影响。

约束和带宽效率。除开销之外，使前向纠错能够消除重传的关键特性在于其从符号丢失中恢复的能力。一旦经网络传输，接收端便获得一定数量的幸存源符号n_{recv} 以及幸存修复符号 k_{recv}。成功解码仅要求接收到的符号总数满足可行性条件：$n_{\text{recv}} + k_{\text{recv}} \geq n$，在此情况下，无论丢失的是哪些具体符号，全部 n 个原始源符号均可被重构。因此，只要冗余足以覆盖缺失符号的数量，前向纠错就能有效掩盖丢包。

在自适应流媒体场景中，冗余需求与网络和播放动态紧密相关。较高的丢包率需要更大的修复预算（k），而有限的链路有效吞吐量或较小的播放缓冲区会限制可容忍的前向纠错开销。为说明这种权衡，我们使用增强版 SABRE-FEC 实现（见第 *NVIDIAAsteroid* [37] 节所述）在 Netflix 5G [41] 网络跟踪记录上，对 NVA 4K 视频进行 HTTP自适应流媒体视频点播（VoD）实验。更多实验细节见第 *4* 节。我们将 无前向纠错与默认的 RaptorQ（RQ）配置进行比较，该配置采用固定参数 n=20， k=10 和 S=64 字节。

图2 展示了三种丢包场景下的视频多方法评估融合（VMAF） [43] 得分。在 0% 丢包率下，前向纠错没有任何益处：默认 RQ 设置的 50% 开销会降低有效带宽，导致 VMAF 相较于无前向纠错下降 10 个单位，约为 2 个最小可觉差[3]（JND） [10]。此处的质量下降完全源于开销，而非恢复。

在 1% 丢包的情况下，无前向纠错和 RQ 都会因符号丢失而出现质量下降，但两者差距有所缩小：RQ 现在比无前向纠错低 7 个 VMAF单位（1 个刚可察觉差异）。尽管前向纠错开始发挥作用，但默认冗余量（k=10）仍会带来过多开销。将修复符号减少至k=5 （25% 开销）可提高 VMAF，使其与无前向纠错的差距低于最小可觉差阈值。进一步减少至 k=1 （5% 开销）后，RQ 的表现略胜于无前向纠错，高出 1 个 VMAF单位。值得注意的是，既可以通过增加 n，也可以通过减少 k来降低开销，两种做法带来的定性影响相似。

然而，这并不意味着带有 k=1 的 RQ 始终是最优选择。在 5% 丢包情况下，同一低冗余配置表现不佳：其性能落后于无前向纠错 14 个 VMAF单位（超过 2 个最小可觉差）。在这一高丢包率区间，冗余不足以弥补符号丢失，解码经常失败，进而触发重传和重新缓冲。因此，冗余不足的表现甚至不如完全不使用前向纠错。

[3]6 个 VMAF单位的差异通常被视为最小可觉差阈值；超过这一阈值时，普通观众便能察觉到质量变化。

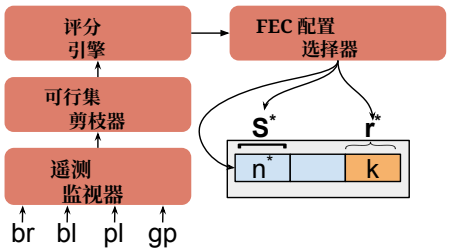

图 3：TAROT 自适应前向纠错调整。

我们还评估了将符号大小 S 增至 128 和 256 字节的影响。将 S 加倍，编码延迟也大致加倍。然而，在播放缓冲区充足的视频点播场景中，这种计算开销并未造成任何可感知的质量下降。这些发现进一步印证了一项关键见解：前向纠错的性能高度取决于冗余、符号化与网络状况之间的相互作用。不存在一种静态配置能在所有场景下都表现出色。因此，前向纠错必须动态调整参数，而不是依赖固定的一刀切设置，才能在网络状况变化时兼顾效率与稳健性。这一关键见解构成了我们所提出解决方案 TAROT 的基础。

3.2 TAROT：自适应前向纠错调整

图 2中的实验分析表明，前向纠错性能对冗余、符号化与网络状况之间的相互作用高度敏感。没有任何一种静态配置能在所有场景下都表现良好。因此，我们提出 TAROT，一种完全自适应的前向纠错控制器，它根据实时网络状况和播放状况，将为每个分段选择最优配置 (n, k, S) 建模为 受约束的优化问题。

3.2.1 问题表述。 前向纠错参数选择问题可形式化为混合整数线性规划（MILP）。尽管该问题可形式化为混合整数线性规划，但候选集 C 的规模有限且易于管理（通常包含数百种配置），因此直接枚举方法在计算上可行，且等同于求解该混合整数线性规划。这避免了通用混合整数线性规划求解器带来的开销。对于每个分段，系统状态由遥测向量 $\mathbf{s} = (br, bl, pl, gp)$表征，其中 br 表示请求的播放比特率（bps）， bl 表示客户端缓冲区占用量（秒）， pl 表示短期数据包丢失概率；而 gp 表示测得的链路有效吞吐量（bps）。

令 C 表示候选前向纠错配置的有限集合，其中每个候选项 $c \in \mathsf{C}$ 都由元组 $(n_c, k_c, S_c,$ 编解码器$_c)$ 定义，其中 $n_c, k_c \in \mathbb{Z}^+$ 以及 $S_c \in \mathbb{Z}^+$（以字节为单位的符号大小）。

我们为每个候选项 $c \in \mathsf{C}$ 引入二元决策变量 $x_c \in \{0,1\}$，其中 $x_c = 1$ 表示是否选择配置 c。混合整数线性规划公式如下：

$$\begin{aligned} \underset{x_c}{\text{minimize}} \quad & \sum_{c \in C} J(c) \cdot x_c \\ \text{subject to} \quad & \sum_{c \in C} x_c = 1 \quad \text{(Single selection)} \\ & \sum_{c \in C} \left(\frac{k_c}{n_c} - \alpha(bl, br) \cdot pl \right) x_c \geq 0 \quad \text{(Feasibility)} \\ & x_c \in \{0,1\}, \quad \forall c \in C \end{aligned} \tag{1}$$

其中，每个候选项的目标函数 $J(c)$ is:

$$J(c) = w_{\text{loss}} P_{\text{loss}}(c) + w_{\text{over}} P_{\text{over}}(c) + w_{\text{blk}} P_{\text{blk}}(c) \tag{2}$$

保护裕度 $\alpha(bl,\ br)$ 根据观测到的丢包情况确定所需的最低冗余比例。这里， B_{eff} 表示客户端有效播放缓冲区水位（以秒为单位）， B_{crit} 是临界缓冲区阈值；低于该阈值时，TAROT 会提高保护程度，以避免即将发生的重新缓冲。其定义如下：

$$\alpha(bl, br) = \alpha_{\min} + \alpha_B \cdot \max(0, B_{\text{crit}} - B_{\text{eff}}) - \alpha_h \cdot \min\left(\frac{gp - br}{br}, h_{\text{cap}} \right) \tag{3}$$

其中， $\alpha_{\min} = 1.0$ 为基准保护因子， $\alpha_B = 0.5$ 会在缓冲区极低时增大余量，而 $\alpha_h = 0.5$ 会在带宽余量充足时减小余量。这既能在缓冲区危机期间提供更强的保护，也能在条件有利时提高效率。自适应权重 $w_{\text{loss}}, w_{\text{over}}, w_{\text{blk}}$经过归一化（$\sum w_\cdot = 1$），并根据s动态调整：High$pl$ 会增大 w_{loss}；low gp 或 tight bl 会增大 w_{over}；critically low bl 会增大 w_{blk}。

3.2.2 TAROT 控制器架构。 虽然混合整数线性规划公式为其提供了理论基础，TAROT 在实际部署中采用高效枚举。如图3所示，TAROT 由四个实现该优化的模块组成。

模块 ***1***：遥测监视器。 对于每个分段请求，TAROT 收集跨层状态向量 $\mathbf{s} = (br、\ bl、\ pl、\ gp)$。这些信号决定了屏蔽丢包所需的最小保护程度，以及系统对开销和延迟的容忍度。

模块 ***2***：可行集剪枝器。 该模块应用混合整数线性规划可行性约束，对候选库 C 进行剪枝。它计算所需的最小冗余比例 $r_{\min} = \alpha(bl、\ br) \cdot pl$并构建可行集 C$f$：

$$C_f = \left\{ c \in C \;\middle|\; \frac{k_c}{n_c} \geq r_{\min} \right\}$$

这实现了混合整数线性规划的约束满足阶段，从而缩小搜索空间。

模块 ***3***：评分引擎。 每个候选项 $c \in Cf$ 均使用公式2中的多目标惩罚模型进行评估。

丢失保护惩罚（P_{loss}）：当修复预算不足以应对预期丢失时，对配置进行惩罚。设 $L_{\text{eff}} = n \cdot pl$为丢失源符号的预期数量：

$$P_{\text{loss}}(c) = \max(0,\ L_{\text{eff}} - \beta \cdot k)^2 \tag{4}$$

其中，编解码效率因子 $\beta \in [0, 1]$ 用于衡量不同前向纠错码族的解码性能。对于里德-所罗门码等最大距离可分码（MDS），$\beta = 1.0$ 因为 k 个修复符号可以恢复任意 k 个丢失符号。像 RaptorQ 这样的近最大距离可分码使用 $\beta = 0.99$，而基于异或的奇偶校验等更简单的编码则使用$\beta < 1.0$，因为它们可恢复的模式有限。这些值根据编解码器规范和经验验证得出。二次形式会严厉惩罚远未达到所需保护水平的配置，使优化过程在面对解码失败时采取审慎策略。这确保 TAROT 优先保证可靠解码，而不是追求微小的带宽节省，因为解码失败会触发代价高昂的重传，其对体验质量的损害比适度增加开销更严重。

开销惩罚（P_{over}）：对超出当前限额的冗余所造成的带宽消耗进行惩罚：

$$P_{\text{over}}(c) = \left(\max\left(0, \frac{k}{n} - \sigma_{\text{free}}(gp, br, bl)\right)\right)^{\alpha_{\text{over}}} \tag{5}$$

开销容许函数 $\sigma_{\text{free}}(gp, br, bl)$ 表示在当前网络状况下可容忍的最大前向纠错开销。其计算方式如下：

$$\sigma_{\text{free}} = \min\left(\sigma_{\text{cap}},\ \sigma_0 + \kappa_B \cdot \max(0, B_{\text{crit}} - B_{\text{eff}}) + \kappa_h \cdot \min\left(\frac{gp - br}{br}, h_{\text{cap}}\right)\right) \tag{6}$$

其中，$\sigma_0 = 0.01$ 为基准开销容许量，$\kappa_B = 0\ 02$ 根据缓冲区缺额（即低于 B_{crit} 的每秒缺额）进行缩放，$\kappa_h = 0.03$ 根据可用带宽余量进行缩放，而 $\sigma_{\text{cap}} = 0.35$ 则设定硬上限。此公式允许在缓冲区较低（需要保护）或带宽余量充足时增加前向纠错开销，同时避免开销过高而降低视频质量。凸惩罚项（其中 $\alpha_{\text{over}} = 1.5$）可抑制过高的开销，以免降低视频质量；而 σ_{free} 容许量会根据当前网络状况进行调整 —— 在缓冲区危机期间或带宽充足时允许增加开销。此机制可防止前向纠错在稳定时段占用过多带宽。

分块惩罚（P_{blk}）：对时序风险施加惩罚。总分块交付时间为：

$$T_{\text{blk}} = \frac{(n+k)S}{gp} + T_{\text{enc}}(n, k, S, \text{codec})$$

$$P_{\text{blk}}(c) = \max\left(0,\ \frac{T_{\text{blk}} - \delta(bl)}{\delta(bl)}\right) \tag{7}$$

截止期限函数 $\delta(bl)$ 定义了在当前缓冲区状态下可接受的最大分块交付时间。其计算方式为 $\delta(bl) = \eta \cdot B_{\text{eff}}$，其中 $\eta = 0.5$，且 B_{eff} 为有效缓冲区水位。这意味着系统可容忍的分块交付时间最高为当前缓冲区时长的 50%，超出后将施加惩罚项。当 $T_{\text{blk}} \geq 2 \cdot \delta(bl)$ 时，惩罚项饱和于 1.0。

超出 $\delta(bl)$ 截止期限的线性相对惩罚提供平滑降级，而非硬约束，使优化器能够在必要时以较小的时序违规换取显著的保护收益。这种灵活性对于应对瞬态网络状况、同时避免突然的质量切换至关重要。

模块 ***4***：***FEC*** 配置选择器. 该模块通过选择得分最低的配置，在可行集上求解简化混合整数线性规划：

$$c^* = (n^*, k^*, S^*) = \arg\min_{c \in C_f} J(c)$$

由此得到该分段的最优冗余率 $r^* = k^*/n^*$。

为评估评分模型的鲁棒性，我们通过调整两个关键超参数进行了小规模敏感性分析：缓冲区感知因子 α_B 和丢包权重缩放因子 λ_p，调整幅度为校准值上下 ±20%。在所有轨迹和两种自适应比特率算法中，平均视频多方法评估融合（VMAF）仅变化了 1–2 个单位，重新缓冲率的变化也不到 0.2%。这表明，TAROT 的性能并非源于脆弱的超参数。相反，多目标公式本身具有结构韧性：其相互竞争的惩罚项（开销、时延、丢包覆盖和截止时间压力）会自然地相互平衡，确保细微的单个权重变化不会显著改变选定的前向纠错配置或其对体验质量的影响。

计算复杂度。枚举方法分为三个阶段：（1）可行性剪枝（$O(|\mathbb{C}|)$ 次比较）；（2）评分（每个候选项执行 15 次操作）；（3）选择（$O(|\mathbb{C}_f|)$ 次比较）。由于 $|\mathbb{C}_f| \leq |\mathbb{C}|$，总体复杂度为 $O(|\mathbb{C}|)$。在典型的 $|\mathbb{C}| \approx 400$ 条件下，使用现代硬件（24 核 CPU，64 GB RAM），决策开销为 $\approx 500\mu s$，与 2-6 秒的分段时长相比微不足道，即使在资源受限的设备上也能满足实时运行要求。

TAROT基于混合整数线性规划的优化在丢包保护、前向纠错开销和数据块交付时延之间取得平衡，避免静态前向纠错设置的脆弱性，同时确保实时部署所需的计算可行性。附录中详细介绍了 TAROT 的设计细节。具体而言，附录 A列出了TAROT所用的完整超参数集，以及其经验校准依据和运行范围。附录 B 包含所有数学公式和形式化证明，包括混合整数线性规划等价性的正确性、全局最优性、约束满足保证、复杂度分析、帕累托最优性、对测量噪声的鲁棒性，以及复合目标函数的稳定性。这些附录共同确立了TAROT 在理论可靠性、最优性和鲁棒性方面的性质，同时使正文能够聚焦于系统设计和实验洞见。

3.3 真实世界部署

尽管我们的评估采用受控设置，但 TAROT 的设计目标是集成到构建于不可靠或部分可靠的 UDP 传输协议之上的真实世界流媒体协议栈中，例如 QUIC [25], MPQUIC [57], RTP/SRT [48], 以及专有的内容分发网络传输协议 [39]。这些传输协议会向应用程序公开数据包传递统计信息，并依赖前向纠错而非重传，因此是 TAROT 的理想部署目标。可靠传输协议（如 TCP、流控制传输协议）能够按序恢复数据，而 TAROT 仍兼容部署在其上方的应用层前向纠错层（如基于 TCP 的 DASH [16]），因为它只需要通用的数据包传递遥测信息，无需传输协议专用的钩子。

实际上，TAROT 对每个分段只需跟踪两个计数器——已发送数据包数和已声明丢失的数据包数——几乎所有媒体传输协议栈都能提供这两项数据。QUIC 和 MPQUIC 通过 qlog [31]导出确认范围和丢包报告；RTP 和 SRT 也会公开类似统计信息；TCP/SCTP 可通过选择性确认块或重传计数器推断丢包情况；基于边缘端的内容分发网络可靠性层则会导出每个数据块的确认/丢失摘要。客户端缓冲区水平、播放比特率和请求时序等跨层指标，可通过通用媒体客户端数据 [12]（Common Media Client Data，CMCD）或通过 [50, 51] 中提出的 CAQ 遥测接口等增强机制获取。这些信号提供了 TAROT 自适应评分模型所需的 br、bl、pl 和 gp 输入。每个分段的短期丢包率按丢失数据包数与已发送数据包数之比计算，并使用指数加权移动平均进行平滑：

$$p = \lambda p_{\text{seg}} + (1 - \lambda) p_{\text{prev}},$$

LLL模式下的值较高 λ，而视频点播的值较低。

TAROT 的逐片段优化刻意保持轻量：评估所有 $(n,\ k,\ S)$ 种配置需要 $\approx 500\,\mu$ 秒，与典型的 2–6 秒分段时长相比微不足道。由于 TAROT 仅依赖标准的数据包传递遥测和广泛

可用的跨层客户端信号，因此能够无缝集成到 DASH/HLS/LLL 播放器、执行实时前向纠错插入的 CDN边缘节点、QUIC/WebRTC 网关以及基于用户数据报协议的超低时延流水线中。总体而言，TAROT 符合现代生产级流媒体系统的可观测性与控制节奏，同时在重传代价高昂或不受欢迎的环境中效果最佳。

4 方法

4.1 SABRE-FEC

SABRE 是一款轻量且广泛使用的模拟器，用于在受控网络状况下评估基于 HTTP 的自适应流媒体逻辑。然而，原始 SABRE 框架仅对吞吐量和往返时间（RTT）的变化进行建模，不支持分组级丢失或任何形式的前向纠错。为了准确研究丢失、冗余与自适应视频流之间的相互作用，我们将 SABRE 扩展为 SABRE-FEC，如图所示4。我们的修改（以灰色突出显示）引入了两个新组件：一个分组丢失仿真模块和一个模块化前向纠错插件。这些新增组件使 SABRE-FEC 能够再现真实的丢失模式，并在客户端的自适应调度循环中评估逐片段前向纠错决策。在 SABRE 中，视频清单仅包含各表示版本的逐片段大小，而不包含实际媒体块。SABRE-FEC 将这些大小用作有效载荷输入，以计算符号化、冗余和前向纠错保护片段大小。

4.1.1 丢包模拟模块。准确建模丢包对于评估任何前向纠错机制都至关重要，但由于丢包的影响会通过传输层动态过程体现出来，因此很难模拟。单次丢包事件就可能触发拥塞窗口崩溃、重传突发、ACK压缩、 pacing中断以及与延迟ACK策略的相互作用。因此，丢包导致的吞吐量下降很少与丢包率呈线性关系，而仅按 $(1-L)$ 缩放带宽的模拟器会严重误判丢包对流媒体失真的影响。

设计原则。如果丢包率为 L，那么传输的数据包中有 L 的比例无法用于提供有效载荷。然而，有效吞吐量受到的影响不仅取决于丢失数据的数量，还取决于传输协议对丢包的反应。基于丢包的拥塞控制算法（例如 CUBIC、Reno），

QUIC恢复和RTP/SRT节奏控制在出现显著丢包时，都会呈现出典型的超线性吞吐量骤降 [8, 22, 38, 40]。 [4, 15, 38] 实证研究反复表明，一旦丢包率超过1–2%，有效吞吐量的下降速度就会快于 $1-L$ 所预示的速度。

设计目标。SABRE-FEC需要一种丢包模型，该模型：(1) 能够捕捉主要的传输层吞吐量崩塌；(2) 保持与协议无关（不进行拥塞窗口建模）；(3) 足够轻量，能够针对每个分段运行一次。

完整的QUIC/TCP恢复仿真开销过高，而完全采用线性近似又不够准确。因此，我们采用一种受经典TCP丢包模型 [2, 4, 5, 24] 以及近期针对QUIC和5G无线接入网行为的丢包敏感性分析 [33, 45, 52]启发、并由解析分析驱动的吞吐量模型。

解析模型。 我们将丢包情况下的有效带宽定义为：

$$B_{\text{loss}} = \frac{B_{\text{link}}}{1 + \gamma\, L \cdot 100 \cdot \sqrt{L}}, \textit{where} \tag{8}$$

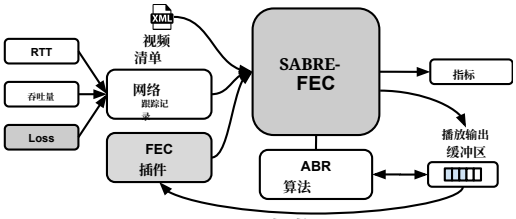

图4：SABRE-FEC架构。

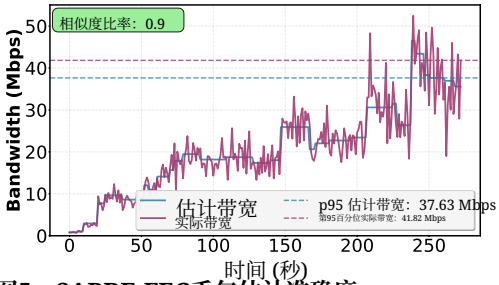

图5：SABRE-FEC丢包估计准确度。

- $B_{\text{链路}}$是跟踪记录中的名义链路带宽，
- L 是丢包率（例如，1% 丢包 $\Rightarrow L = 0.01$），
- γ 是缩放参数（默认值为 $\gamma = 0.5$）。

值得注意的是，γ 是通过与从 Vegvisir 仿真器获得的实际测量结果进行校准后，根据经验选定的，如图5所示。该 $L\sqrt{L}$ 项提供了超线性衰减，反映了已确立的传输行为：

- 在丢包率极低时，吞吐量大致跟随（$1-L$），
- 当丢包率超过 1–3% 时，有效吞吐量会急剧下降，这反映了 TCP 退避，
- 在高丢包率下，该模型趋近于无线链路中出现的非线性崩溃。

该模型无需 ACK 处理、无需拥塞窗口仿真，也无需协议插桩，因此适用于大规模逐分段实验，同时仍能捕捉真实传输行为的定性特征。

与协议无关的解释。 与特定传输协议恢复机制的模型不同，公式（8）可直接应用于：

- QUIC 和 MPQUIC（通过 qlog 暴露的丢包信息），
- 具有显式丢包计数器的 RTP/SRT/WebRTC 传输，
- 采用应用层丢包估计的 TCP 上的 DASH/HLS

这使丢包模拟在所有评估方案——无前向纠错、静态前向纠错和TAROT——之间保持一致。我们指出，尽管不同的拥塞控制算法（例如 TCP CUBIC/Reno 与 QUIC BBR）对数据包丢包的响应各不相同，我们的模型仍有意使用单个经经验校准的缩放因子来捕捉主要的丢包—有效吞吐量崩塌趋势；γ 的特定协议校准留待未来工作。

为验证丢包—有效吞吐量公式的准确性，我们将公式（8）与从以下工具获得的实际测量结果进行了比较：*Veg-visir* [19], 这是一款开源网络仿真器，将`tc netem`与容器化客户端/服务器端点相结合。Vegvisir 能够在受控网络劣化下提供真实的数据包传递统计，同时保留真实的 TCP/QUIC 拥塞控制动态。我们重放了 50 多条 LTE 和 5G 轨迹 [30, 41, 42, 56] 并注入了

受控的 5% 数据包丢失配置，同时运行基于吞吐量的自适应码率。图5比较了*Vegvisir* 客户端测得的实际有效吞吐量（“观测吞吐量”）与 B_{loss} 公式（8）预测值。在所有网络状态下，解析模型的误差均处于模拟器的**第95百分位误差范围**内。此外，我们计算得出的相似度比率为 0.9，其定义为：在所有轨迹中，预测有效吞吐量与观测有效吞吐量之间的偏差小于第95百分位误差界限的时间样本所占比例。这种一致性表明，该模型能够捕捉丢包对传输有效吞吐量的主要影响，同时避免完整协议模拟的开销。因此，该模型足够准确，可用于驱动TAROT 的分段级前向纠错决策；同时也足够轻量，可在自适应流媒体的紧凑控制环路中在线运行。

在吞吐量非线性骤降的情况下，静态前向纠错和自适应前向纠错的表现截然不同。准确建模丢包导致的有效吞吐量下降至关重要；否则：

- 静态 FEC 方案会显得异常强劲（因为实际带宽损失被低估），并且
- 自适应方案会显得不稳定（因为冗余–缓冲区–有效吞吐量之间的耦合关系受到扭曲）。

我们的解析丢包模块确保 SABRE-FEC 能够准确捕捉丢包带来的定性影响，从而实现对 TAROT 公平且真实的评估。

4.1.2 模块化前向纠错插件。 SABRE-FEC 包含一个模块化前向纠错插件，可通过统一接口评估不同的编码族。该模块涵盖前向纠错的两项基本成本：(i) 传输修复符号的带宽成本 ，以及 (ii) 编码和解码的计算成本 。这些成本会直接影响自适应码率吞吐量估算器，因此必须对其进行明确建模。

为如实反映这些相互作用，前向纠错层会计算调整后的分段传输时间，其中既考虑冗余导致的传输字节数增加，也考虑生成修复符号所需的额外时间。编码会带来延迟，使分段无法立即开始传输；这一延迟会明确计入分段下载时间。

在接收端，我们采用与现代 DASH、HLS 和 LL-HLS 播放器一致的标准假设：解码在非阻塞后台线程上进行。在此类架构中，解码延迟不会 影响测得的吞吐量或自适应码率对分段获取时间的估算；它只会影响渲染流水线。在我们的实验中，RaptorQ、里德–所罗门和异或的解码延迟仅为几毫秒，远低于典型的缓冲区水位（直播流媒体为 6 秒，点播流媒体为 60 秒），因此对播放输出的影响可以忽略不计。在这一假设下，只有编码延迟处于关键路径上，必须纳入吞吐量模型。

我们将三种广泛使用的前向纠错码族集成到 SABRE-FEC 中：

- **异或（块校验）：** 编码成本极低；仅支持单丢失恢复。
- **里德–所罗门（RS）：** 具有强恢复保证的最大距离可分码，但编码成本较高。
- **RaptorQ（请求）：** 支持灵活块大小且计算成本适中的喷泉码。

TAROT的自适应前向纠错调整（如图3所示）也在此模块中实现。为真实地参数化编码延迟，我们对开源的quic-go-fec 实现 [32, 36] 进行了基准测试，并测量了不同(n， k， S）设置下每个数据块的平均编码时间。值得注意的是，测量结果显示，RaptorQ 的平均编码成本约为每字节 22 纳秒，而里德–所罗门约为每字节 35 纳秒。我们将这些数值输入 SABRE–FEC，以模拟真实的计算开销，并确保编码延迟的影响反映在片段获取时间中，从而模拟现实环境中的行为。

前向纠错开销下的带宽缩减。设 n 表示源符号数量， k 表示修复符号数量，S 表示符号大小。前向纠错引入的带宽膨胀由以下开销因子表示：

$$\text{overhead_factor} = \frac{n+k}{n}.$$

前向纠错模块会根据 (i) 传输冗余和 (ii) 修复符号缓解丢包的程度，调整向自适应码率逻辑提供的有效带宽。由此得到的有效载荷吞吐量为：

$$B_{\text{FEC}} = \frac{B_{\text{link}}(1 - l_{\text{eff}})}{\text{overhead_factor}}, \tag{9}$$

其中， l_{eff} 表示应用该编码的恢复能力后残余丢失，其计算方式为：

$$l_{\text{eff}} = \begin{cases} L\left(0.4 + 0.6\left(1 - \frac{\text{headroom}}{\text{coverage}}\right)\right), & L \le \text{coverage}, \\ L - 0.8\,\text{coverage}, & L > \text{coverage}, \end{cases}$$

截断至 $[0, L]$，其中 L 是观测到的丢包率，覆盖率 $= \frac{k}{n+k}$。该模型捕捉了冗余收益递减的特点：当覆盖率显著高于丢包率时，前向纠错可抑制大部分丢包；当覆盖率不足时，则只能缓解其中一部分。因此，公式（9）表示自适应码率算法可见的有效载荷有效吞吐量。结合编码时延，SABRE-FEC 以符合实际运行的流媒体系统的方式，同时对带宽膨胀和计算开销进行建模。

由于前向纠错插件提供统一的 API，用于计算经前向纠错保护的吞吐量、残余丢失和编码时延，TAROT可以集成任何前向纠错方案——异或、RS 或 RQ——而无需修改。如第4.3节所示，TAROT通过动态选择(n， k， S)，在冗余、分块时延和计算成本之间取得平衡，从而在所有三种码族中持续提升体验质量。

4.2 仿真设置

4.2.1 视频内容与参数。 对于 TAROT 的所有在线评估，我们使用以高级视频编码编码的 *NVIDIAAsteroid* (NVA) 视频数据集 [37], [58]，其中包含十种码率表示，范围从 288p@0.5 Mbps 到 2160p@40 Mbps (4K)。实验在两种流媒体场景下进行：视频点播 (VoD)，采用 4 秒片段和 60 秒的最大播放缓冲区；以及低延迟直播 (LLL)，采用 2 秒片段和 6 秒的播放缓冲区。每次实验都会流式传输完整的 4 分 30 秒内容，以确保捕捉到长期自适应行为，并使稳态性能成为主要表现，而非瞬态启动效应。

4.2.2 网络轨迹。 为了对 TAROT 进行在线评估，我们采用了四种广泛使用的真实网络轨迹：*Netflix 5G*轨迹 [41], 、比利时 *LTE* 轨迹 [56], 、*Amazon5G* 轨迹 [41],以及*Cascade* 轨迹 [20] ，以模拟真实的 5G↔4G 切换

表 1：网络跟踪数据。

网络跟踪	平均吞吐量 (Mbps)	标准差 (Mbps)
Netflix5G [41]	33	18
Amazon5G [41]	25	10
比利时 LTE [56]	20	5
Cascade [20]	30	15

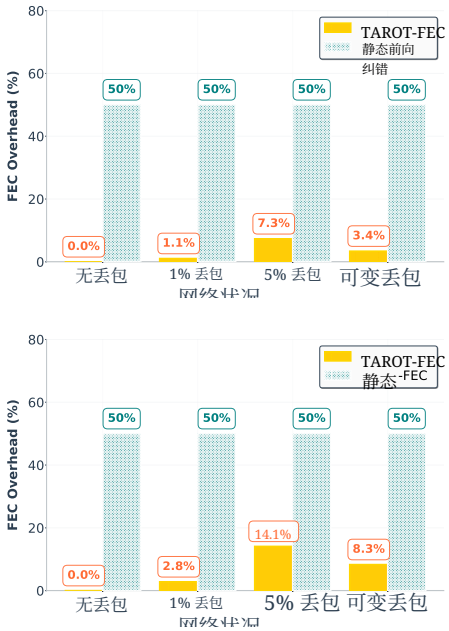

图 6：在 LLL 模式（上）和 VoD 模式（下）下，对多种网络轨迹和自适应码率算法的静态前向纠错与 TAROT-FEC 开销进行比较，取平均值。

动态特征。这些轨迹的特征总结于表1中。每条轨迹均在四种丢包配置下进行评估：(i) 无丢包，(ii) 稳定的 1% 丢包率，(iii) 稳定的 5% 丢包率，以及 (iv) 从 0–5% 范围内的随机采样器中抽取的可变丢包模型，用于捕捉拥塞的蜂窝网络中观察到的波动性。

4.2.3 自适应码率算法。 我们使用原始 SABRE [53] 框架提供的两种自适应码率方案评估 TAROT：(i) 基于吞吐量的 (THR) 自适应码率算法，根据近期下载速率选择可持续的最高比特率；(ii) 动态 [55]自适应码率算法，结合吞吐量估算与基于缓冲区感知的 BOLA 式决策，具体如 SABRE 仿真器所述。在 VoD 配置中，两种自适应码率算法均正常运行，因为播放缓冲区最多可增长至 60 秒，使动态自适应码率算法能够利用其基于缓冲区的逻辑。然而，在 LLL 配置中，缓冲区上限仅为 6 秒。在如此严格的缓冲区限制下，动态自适应码率算法因其基于 BOLA 的效用函数而反复降至最低质量等级。因此，对于 LLL，我们仅报告 THR 自适应码率算法的结果；而在 VoD 模式下，则评估两种自适应码率算法。

4.2.4 前向纠错策略。 为严格评估 TAROT 在不同编码族中的表现，我们考察集成于 SABRE-FEC 框架中的所有前向纠错机制：(1) **无前向纠错**，(2) **RaptorQ（RQ）**，(3)**里德–所罗门（RS）**，(4) **TAROT增强型 RaptorQ （RQ*）**，以及 (5) **TAROT增强型里德–所罗门 （RS*）**。

我们不纳入基于学习的前向纠错方案（例如 RL-FEC、RL-AFEC），因为这些方法仅关注冗余率调优且需要预训练，而 TAROT 的优化驱动设计

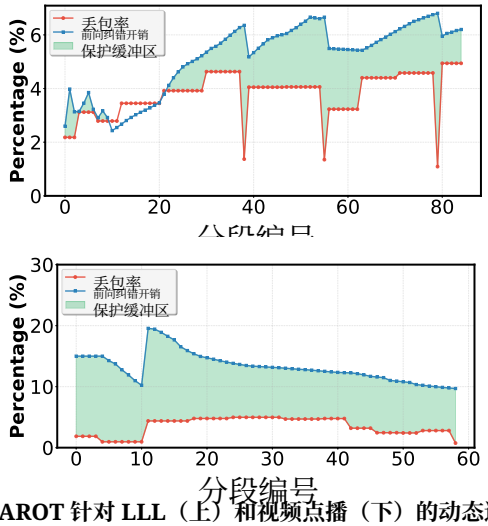

图7：TAROT 针对 LLL（上）和视频点播（下）的动态逐分段前向纠错调整。“保护缓冲区”表示 TAROT 为每个分段选择的冗余度，以前向纠错块中保护符号所占的比例表示（$r = k/n$）。

能够联合选择完整的 (n，k，S) 配置，并在 SABRE-FEC 中在线运行，无需任何预先训练，因此与这些方法进行比较并不公平。更广泛地说，基于学习的前向纠错方案作为黑盒控制器运行，可解释性有限，也不提供明确的回退保证；而 TAROT 的白盒优化会明确揭示并权衡丢包保护、开销和时序约束之间的取舍。RQ 和 RS 均采用相同的默认前向纠错参数（n=20，k=10，S=64），这些参数取自公开提供的前向纠错实现 [32, 35]，，我们已将其集成到 SABRE-FEC 中。我们有意不将基于异或的前向纠错纳入结果。异或采用固定的 2 源/1 校验配置：(i) 不提供可调的 (n，k，S) 变量，(ii) 带来固定的 50% 开销，且 (iii) 仅支持单符号恢复。因此，异或在结构上与 TAROT 的自适应参数选择不兼容，因为它没有可供优化的有效自由度。此外，初步实验表明，由于恢复能力较弱且开销较高，在所有轨迹和丢包率下，异或的表现始终不如 RQ 和 RS。因此，排除异或有助于对能够从自适应优化中获益的 FEC 方案进行更清晰、更公平的比较。

4.2.5 性能指标。 我们使用广泛采用的性能指标评估 TAROT：(1) 视频多方法评估融合（VMAF）[43]，，(2) 重缓冲时长（RD），以及 (3) 前向纠错开销，即相对于原始有效载荷大小所传输的额外修复字节的比例。

4.2.6 系统配置。所有实验均在搭载 24 核 CPU、64 GB 内存的 Ubuntu 22.04.3 系统上运行，使用 Python 3.11。

4.3 结果与讨论

4.3.1 TAROT 对前向纠错开销的影响。如图6所示，TAROT-FEC 在 VoD 和 LLL 设置下，只要未观测到丢包，就会完全抑制冗余。这一行为源于TAROT 的设计：只有在丢包估计器检测到非零数据包丢失时，才会启用前向纠错，因此该机制本质上是按需响应，而非始终开启。

表 2：LLL 和 VoD 模式下，所有轨迹和自适应比特率算法的质量—开销权衡平均值。此处，r 表示前向纠错开销。

Mode	Loss	无丢包			1% 丢包			5% 丢包			可变丢包		
	前向纠错类型	视频多方法评估融合	率失真	r(%)	视频多方法评估融合	率失真	r(%)	视频多方法评估融合	率失真	r(%)	视频多方法评估融合	率失真	r (%)
LLL	无前向纠错	**60.61**	1.51	0	**59.58**	1.71	0	50.05	1.26	0	55.05	1.49	0
	RQ	50.65	1.05	50	50.11	0.88	50	50.27	1	50	49.86	0.78	50
	RS	50.54	0.78	50	50.45	1.29	50	49.75	1.2	50	49.34	1.18	50
	RQ*	**60.95**	1.5	0	**60.14**	1.42	1	**59.71**	1.22	7.05	**62.77**	1.36	3.4
	RS*	**60.92**	1.5	0	**59.92**	1.84	1.2	**59.71**	1.45	7.55	**62.60**	1.37	3.4
VoD	无前向纠错	**78.97**	0.35	0	**75.98**	0.12	0	62.13	0.08	0	70.93	0.14	0
	RQ	65.46	0.03	50	64.15	0.04	50	62.80	0.07	50	62.63	0.04	50
	RS	62.52	0.04	50	63.17	0.04	50	62.68	0.08	50	64.05	0.05	50
	RQ*	**79.50**	0.37	0	**78.33**	0.14	2.75	**71.90**	0.07	14.2	**76.03**	0.01	8.1
	RS*	**79.35**	0.13	0	**79.84**	0.19	2.85	**75.69**	0.08	14	**78.04**	0.02	8.5

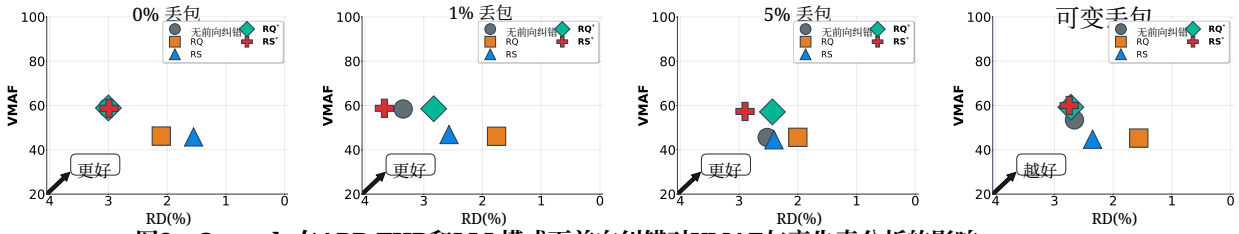

图8：Cascade在ABR:THR和LLL模式下前向纠错对VMAF与率失真分析的影响

一旦路径无丢包，静态前向纠错仍会注入冗余，造成不必要的开销。无论引入的是恒定 1% 或 5% 丢包率，还是在 0% 至 5% 之间变化的可变丢包率曲线，TAROT 均可将 LLL模式下的开销最多降低 46%，将 VoD模式下的开销最多降低 41%。这得益于其多目标优化，它会选择足以满足观测丢包所要求的恢复预算的最小 $(n,\ k,\ S)$ 配置。图7展示了在基于吞吐量的自适应码率下，Amazon 5G 跟踪记录中 5%丢包场景下的这种自适应行为。在整个会话期间，TAROT能够及时响应丢包波动，维持足够的恢复覆盖率，并避免静态前向纠错产生的过量冗余。

这些开销的节省直接转化为体验质量的提升：由于修复字节更少，更多链路带宽可用于视频片段，使自适应码率能够在LLL和VoD模式下选择质量更高的表示版本（详见第 4.3.2 节）。值得注意的是，VoD模式下的节省更为显著，这是因为其分段时长较长，而较长的分段时长原本会放大静态冗余的成本。

4.3.2 TAROT对HTTP自适应流媒体视频流传输的影响。

▸**LLL模式**。图 8 展示了在不同丢包配置下，LLL模式中THR自适应比特率对Cascade跟踪记录的无前向纠错、静态RQ和RS基线，以及其 TAROT增强版本（RQ* 和RS*）之间的对比。在无丢包场景下，由于网络未出现 数据包丢失，TAROT 从未针对RQ* 和RS*触发，因此这些自适应版本的表现与无前向纠错相似。

相比之下，静态RQ和响应方案被锁定在其固定的（n=20, k=10, S=64）配置中，即使不需要保护，也会带来50%的开销。这种无条件开销抑制了暴露给自适应码率逻辑的有效带宽，并迫使播放器在整个会话期间选择较低质量的表示。因此，静态RQ和响应落后于无前向纠错和TAROT增强变体13个VMAF单位（约2个最小可觉差）。它们略低的重缓冲比率（约1%）仅仅是始终以较低比特率进行流媒体传输的结果；这种微小的降低远不足以抵消因开销膨胀而造成的巨大感知质量损失。

在1% 丢包场景下，我们观察到类似的趋势。由于丢包程度仍然较轻，无前向纠错、请求*和响应* 取得了几乎无法区分的视频多方法评估融合，而静态RQ和RS基线则因其固定的50%开销继续落后约12个VMAF单位（2个最小可觉差）。TAROT确实会对测得的丢包作出反应并激活其估计器，但1%丢包下所需的保护远小于50%。因此，TAROT选择轻量级前向纠错配置，其开销保持接近于零，从而使其能够保持与无前向纠错几乎相同的有效吞吐量和表示选择。相比之下，无前向纠错无法通过前向纠错掩盖丢包，且1% 丢包下的端到端重传延迟与轻度前向纠错开销和编码延迟的综合代价相当。因此，在该情形下，无前向纠错、请求*和响应* 实现了基本相同的体验质量权衡，并且所有方案都表现出相似的重缓冲比率。

在5%丢包场景下，请求* 和响应* 优势明显，其成绩比无前向纠错、响应和请求高出12个VMAF单位（2个最小可觉差）。这一改进源于TAROT识别出抵御升高丢包率所需的最小开销，仅施加刚好足以防止重传导致的卡顿的保护，同时保留大部分可用带宽。相比之下，无前向纠错在5%丢包时急剧劣化，因为重传变得频繁且代价日益高昂，从而导致大幅质量下降。令人惊讶的是，静态RQ与响应方案即便在这一高丢包率区间也未能超越无前向纠错，表现与之相当。尽管前向纠错在5%丢包时应带来明显益处，但其固定的50%开销严重抑制了有效吞吐量，使得两种静态方案尽管传输了显著更多的冗余，仍落后于TAROT增强变体近12个VMAF单位。

在丢包率在0%到5%之间波动的可变丢包场景中，我们观察到与固定5%丢包条件下类似、但略窄的性能差距。尽管丢包率存在波动，RQ* 和RS* 仍比无前向纠错高出6个视频多方法评估融合单位（1个刚可察觉差异）。这一增益再次体现了

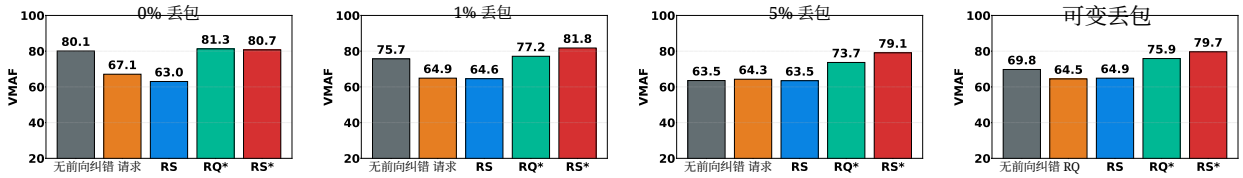

图9：在 Netflix 5G 上，ABR:THR 和模式：视频点播下，前向纠错对视频多方法评估融合与率失真分析的影响。

TAROT能够动态调整开销，以匹配瞬时丢包率，仅在需要时提供保护。相比之下，静态RQ和RS方案的表现仍然较差：它们比无前向纠错低约8个视频多方法评估融合单位（1个刚可察觉差异）。与之前一样，它们固定的50%开销会在整个会话期间降低有效吞吐量，即使在丢包率较低、如此高的开销毫无益处的时段也是如此。因此，这两种静态前向纠错方案仍无法利用丢包率降低的时段，其感知质量也持续低于无前向纠错和TAROT增强变体。

值得注意的是，其余网络轨迹呈现出与 Cascade 轨迹相同的定性行为。为简洁起见，我们在表2中报告了 LLL 模式下的跨轨迹平均值，并比较了无前向纠错、静态前向纠错（RS、RQ）以及我们的 TAROT增强变体（RQ* 和 RS*）。汇总结果进一步印证了三个关键发现：（i）TAROT在稳定、无丢包条件下具有零开销优势；（ii）在中等丢包情况下，它能够以极少的冗余实现保护与吞吐量之间的平衡；（iii）在高丢包场景下，它具有更强的韧性，自适应前向纠错可显著提升视频多方法评估融合得分，同时开销比静态方案低一个数量级。

▸**视频点播模式**。 图 9 比较了无前向纠错、静态RQ和RS基线，以及它们的 TAROT-增强版本在所有丢包配置下、THR自适应比特率算法使用 *Netflix 5G* 跟踪记录时的表现。与LLL模式不同，由于播放缓冲区大得多（60秒），视频点播模式下所有方案和所有丢包条件均未出现任何 重新缓冲事件。因此，我们仅报告视频多方法评估融合结果。视频点播模式下的趋势与LLL模式相似，但视频多方法评估融合差异更加 显著 。在多种丢包配置下，静态RQ和RS的表现比无前向纠错及其 TAROT- 增强版本低了 12 个以上的视频多方法评估融合单位（约相当于2个最小可觉差）——视频点播模式较高的运行码率进一步拉大了这一差距。由于视频点播模式拥有充足的缓冲区，自适应码率算法会迅速稳定在高质量表示版本上，因此，前向纠错开销成为主导因素：静态RQ/RS固定的50%冗余度对有效吞吐量的抑制远比LLL模式下严重，并大幅降低视频多方法评估融合值。

表2中视频点播的跨轨迹平均值证实了相同的在 Netflix 5G 跟踪数据中观察到的模式：在稳定和中等丢包情况下，TAROT 始终保持较高的视频多方法评估融合值；在高丢包情况下，其鲁棒性显著更优，同时开销仅为静态前向纠错的一个数量级。这些发现凸显了 TAROT 自适应的逐分段 FEC参数选择的重要性。在 LLL 和视频点播模式下，TAROT 都能带来相同的核心优势：低丢包时零成本，中等丢包时开销极小，高丢包时提供更优的保护；这直接提升了体验质量，较高的视频多方法评估融合分数也证明了这一点。

4.3.3 为什么仅靠冗余还不够？图10 比较了TAROT-前向纠错与一种 R-FEC 控制器的表现，后者仅调整冗余率，同时保持(n, S) 不变。尽管 R-FEC 在 LLL模式下优于静态前向纠错（表2），但表现仍逊于TAROT-前向纠错

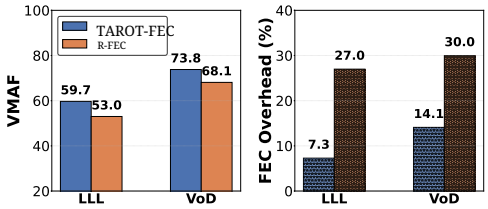

图10：消融实验：TAROT 与仅使用冗余的控制器（R-FEC）对比，在所有轨迹、自适应比特率算法和 FEC 方案上取平均，丢包率为 5 %。

约 6 个 VMAF单位（1 个刚可察觉差异）。在视频点播中，差距有所缩小，但在感知上仍有显著意义：尽管TAROT-前向纠错相比静态前向纠错提升近 2 个刚可察觉差异，R-FEC 仅弥补约 1 个刚可察觉差异。其原因在于结构限制——仅靠冗余无法满足前向纠错块配置所施加的时序与恢复约束。R-FEC（图10）在两种模式下的平均冗余量都比 TAROT-前向纠错高出 18%，但感知质量却更低。有效的丢包保护取决于(n, k, S)的共同作用，这些参数决定了编码延迟、修复能力和每个数据块的传输成本。 TAROT 会调整完整配置，而不仅仅是冗余，因此在各种网络状况下都能持续实现更高的体验质量。

5 局限性与未来工作

尽管 TAROT 在各种轨迹和自适应比特率算法中均表现出良好效果，但仍存在若干局限。首先，我们的丢包—有效吞吐量公式能够捕捉主要的超线性崩塌现象，但对特定传输协议的行为进行了抽象处理（如拥塞窗口演变、确认时序和发送节奏伪影）。尽管 SABRE-FEC 提供了可控且可重复的条件，但它仍然是一个仿真框架；此外，我们的可变丢包模型虽然可作为网络波动的一阶近似，能够捕捉短期波动，却未明确表示移动网络、Wi-Fi 和卫星链路中常见的相关突发丢包。未来一项重要工作是将 TAROT 扩展到突发丢包过程（吉尔伯特–埃利奥特模型 [34]），并在完整的 QUIC/TCP 仿真或硬件测试平台中验证其行为。

其次，编码延迟采用基于软件的 Rap-torQ 和里德–所罗门实现进行建模。硬件加速或 SIMD 优化编码器可能会改变时延与开销之间的权衡，这表明需要进行设备特定性能分析和自适应复杂度控制。同样，TAROT 目前以分段粒度进行自适应；超低时延工作负载（例如亚秒级直播游戏或 WebRTC 流水线）可能会受益于块级或帧级 FEC 决策。第三，TAROT 有意采用内容无关的设计：它对所有片段应用相同的保护策略，不分析编解码器结构或帧重要性。这样可以使控制器保持轻量，避免在边缘端进行视频解析，并确保其适用于各种编解码器和传输格式。然而，从感知角度来看，这种设计本质上并非最优——丢失一个

I 帧可能会破坏整个 GOP，而 P/B 帧中的错误影响要小得多。若不区分这些差异，TAROT 可能会过度保护不太关键的数据，却保护不足视觉上最重要的片段。纳入内容感知信号（例如帧类型、运动复杂度和显著性）是提升感知效率的一个有前景的方向。

第四，尽管 TAROT 通过优化驱动的自适应展现出优异性能，但我们有意避免采用机器学习组件，以确保可证明的最优性保证、最低的计算开销（≈500μ秒）以及无需训练数据即可立即生效。不过，未来工作可以探索混合方法，在 TAROT 的优化核心中加入学习组件，例如使用 LSTM 网络预测丢失模式，或通过强化学习根据长期体验质量目标调整权重参数。

最后，我们的评估侧重于 LTE/5G 移动网络轨迹。Wi-Fi 6/7 竞争、毫米波移动性、卫星链路或骨干网拥塞等更广泛的环境可能呈现不同的时变丢失结构。将 TAROT 扩展至这些环境，并探索更深入的跨层协作（例如主动队列管理、RAN 排队信号），是未来部署的一个颇具前景的方向。

6 结论

我们介绍了 TAROT，这是一种由优化驱动的 FEC 控制器，可通过轻量级多目标评分模型，针对每个分段选择冗余度、符号化方式和块大小。TAROT 能根据实时丢包、有效吞吐量和缓冲区状态进行自适应调整，避免静态 FEC 方案的僵化。为支持真实场景下的评估，我们开发了 SABRE-FEC，这是 SABRE 的高保真扩展，纳入了非线性丢包–有效吞吐量模型，以及带有实测编码成本的模块化 FEC 层。在 LTE、5G 和切换轨迹中，无论是 LLL 还是视频点播场景，TAROT 都能持续降低开销（最高达 45%）、提升感知质量（相较于无前向纠错提升 +10%，相较于静态 FEC 提升 +20%），并减少重新缓冲，而每个分段的决策成本仅为 $\approx$ 500 μ 秒。TAROT 仅依赖标准的数据包传递遥测，因此无需修改协议，即可集成到 DASH/HLS 客户端、CDN 边缘节点或基于 QUIC 的传输栈中。除了支持 TAROT 外，我们的 SABRE-FEC 框架还可作为可复用的基准测试工具，用于评估未来的 FEC 方案：它提供了一个受控的高保真测试环境，可在真实的丢失模式和 ABR 动态下，快速测试冗余策略、码族和参数调优逻辑。总之，TAROT 为现代视频流媒体的自适应、优化驱动型 FEC 参数调优提供了一条实用而有效的路径，使系统能够在多变的网络状况下提升体验质量，并更高效地利用带宽。

致谢。本研究由加拿大自然科学与工程研究委员会（NSERC）发现基金 RGPIN-2023-04744，以及 NSERC Alliance (ALLRP 586434-23)-Mitacs Accelerate (IT36792) 项目资助。

A TAROT 模型超参数 RS

表3 列出了TAROT 自适应前向纠错调整所用的全部超参数。所有超参数均基于由 10% 可用网络轨迹组成的验证集进行经验校准。缓冲区阈值（$B_{\text{sat}} = 6.0$s，$B_{\text{crit}} = 3.0$s）与典型分段时长和关键播放限制相符。保护因子（$\alpha_{\min} = 1.0$, $\alpha_B = 0.5$ $\alpha_h = 0.5$ 1 1 ，）可确保基线保护，同时根据缓冲区和带宽条件进行调整。开销参数（$o_0 = 0.01$, $k_B = 0.02$ $k_h = 0.03$ ，）允许维持最低基准开销，并在不利条件下进行受控增加。权重自适应参数在多种网络场景下，兼顾丢包保护、开销控制和时序约束。算法1 概述了TAROT 自适应逐分段前向纠错调整的整体流程。

B 数学证明

B.1 混合整数线性规划公式正确性的证明

定理B.1（混合整数线性规划等价性）。算法*1*中的枚举方法求解的问题与方程（*1*）中的混合整数线性规划公式所表示的优化问题相同。

证明。令 Cf 为经过约束剪枝后的可行集。混合整数线性规划目标函数为：

$$\min \sum_{c \in C} J(c) x_c \quad \text{subject to} \quad \sum_{c \in C} x_c = 1, \quad x_c \in \{0, 1\}$$

由于恰有一个 $x_c = 1$，其余均为 0，因此目标函数可简化为所选配置对应的 $J(c)$。算法1对所有 $c \in$ Cf 评估 $J(c)$，并选择 arg min，这等价于在剪枝后的可行集上求解混合整数线性规划。

表 3：TAROT 超参数值。

参数	值
缓冲区管理	
B_{sat}	6.0 秒
B_{crit}	3.0 秒
h_{cap}	2.0
丢包保护	
$\alpha_{\min}$	1.0
α_B	0.5
α_h	0.5
开销控制	
o_0	0.01
k_B	0.02
k_h	0.03
o_{cap}	0.35
α_{over}	1.5
分块控制	
η	0.5
$\text{hardcap}_{\text{tblk}}$	1.5
权重自适应	
$w_{\text{loss}}^{\min}$	0.5
λ_p	6.0
p_{cap}	0.15
$w_{\text{over}}^{\min}$	0.5
λ_B	0.5
λ_h	0.4
$w^{\min}$ 数据块	0.3
λ_{risk}	0.6
λ_{hneg}	0.6

约束 $\sum_{c\in C}(\frac{k_c}{n_c} - \alpha\cdot pl)x_c \geq 0$ 通过可行性剪枝来强制执行，确保仅考虑满足 $\frac{k_c}{n_c} \geq \alpha\cdot pl$ 的配置。□

Algorithm 1 TAROT: Optimal FEC Parameter Selection per Segment.

Require: loss pl, buffer bl, goodput gp, playback bitrate br, payload size P, candidate set $C = \{(n, S, r, \text{codec})\}$

$C_{\text{feasible}} \leftarrow \emptyset$
$B_{\text{eff}} \leftarrow \min(\max(bl, 0), B_{\text{sat}})$
$h_{\text{raw}} \leftarrow (gp - br)/\max(br, \epsilon)$
$h \leftarrow \min(\max(h_{\text{raw}}, -10.0), 10.0)$
$h_+ \leftarrow \max(0, h)$, $h_- \leftarrow \max(0, -h)$
▷ Compute protection margin $\alpha(bl, br)$
$\alpha \leftarrow \alpha_{\min} + \alpha_B \cdot \max(0, B_{\text{crit}} - B_{\text{eff}}) - \alpha_h \cdot \min(h_+, h_{\text{cap}})$
$\alpha \leftarrow \max(0.5, \alpha)$
for each (n, S, r, codec) in C **do**
  $k \leftarrow \lceil rn \rceil$, $T \leftarrow n + k$, $o \leftarrow k/n$, $cov \leftarrow k/T$
  ▷ Feasibility check
  if $o < \alpha \cdot pl$ **then continue**
  end if
  ▷ Loss effectiveness $f_{\text{loss}}(pl, cov)$
  if $pl \leq cov$ **and** $cov > 0$ **then**
    $headroom \leftarrow cov - pl$
    $reduction \leftarrow headroom/cov$
    $l_{\text{eff}} \leftarrow pl \cdot (0.4 + 0.6 \cdot (1 - reduction))$
  else
    $l_{\text{eff}} \leftarrow pl - 0.8 \cdot cov$
  end if
  $l_{\text{eff}} \leftarrow \max(0, \min(l_{\text{eff}}, pl))$
  ▷ Recalculate headroom with FEC effects
  $G_{\text{payload}} \leftarrow gp \cdot (1 - l_{\text{eff}})/((1 - pl) \cdot (1 + o))$
  $h \leftarrow (G_{\text{payload}} - br)/\max(br, \epsilon)$
  $h \leftarrow \min(\max(h, -10.0), 10.0)$
  $h_+ \leftarrow \max(0, h)$, $h_- \leftarrow \max(0, -h)$
  ▷ Overhead penalty
  $o_{\text{free}} \leftarrow o_0 + k_B \cdot \max(0, B_{\text{crit}} - B_{\text{eff}}) + k_h \cdot \min(h_+, h_{\text{cap}})$
  $o_{\text{free}} \leftarrow \max(0, \min(o_{\text{free}}, o_{\text{cap}}))$
  $P_{\text{over}} \leftarrow \max(0, o - o_{\text{free}})^{\alpha_{\text{over}}}$
  ▷ Blockization penalty
  $t_{\text{blk}} \leftarrow (8 \cdot T \cdot S)/\max(gp, \epsilon)$
  if $t_{\text{blk}} > \text{hardcap}_{\text{tblk}} \cdot B_{\text{eff}}$ **then continue**
  end if
  $P_{\text{blk}} \leftarrow \min(1.0, \max(0, t_{\text{blk}}/(\eta \cdot B_{\text{eff}}) - 1))$
  ▷ Loss penalty with codec efficiency
  $\beta \leftarrow \text{getCodecEfficiency}(\text{codec})$
  $P_{\text{loss}} \leftarrow \max(0, n \cdot pl - \beta \cdot k)^2$
  ▷ Adaptive weights
  $w_{\text{loss}} \leftarrow w_{\text{loss}}^{\min} + \lambda_p \cdot \min(pl, p_{\text{cap}})$
  $w_{\text{over}} \leftarrow w_{\text{over}}^{\min} + \lambda_B \cdot (B_{\text{eff}}/B_{\text{sat}}) + \lambda_h \cdot \min(h_+, h_{\text{cap}})$
  $w_{\text{blk}} \leftarrow w_{\text{blk}}^{\min} + \lambda_{\text{risk}} \cdot \max(0, 1 - B_{\text{eff}}/B_{\text{crit}}) + \lambda_{hneg} \cdot h_-$
  ▷ Normalize and compute final score
  $sum \leftarrow w_{\text{loss}} + w_{\text{over}} + w_{\text{blk}}$
  $w_{\text{loss}} \leftarrow w_{\text{loss}}/sum$, $w_{\text{over}} \leftarrow w_{\text{over}}/sum$, $w_{\text{blk}} \leftarrow w_{\text{blk}}/sum$
  $J \leftarrow w_{\text{loss}} \cdot P_{\text{loss}} + w_{\text{over}} \cdot P_{\text{over}} + w_{\text{blk}} \cdot P_{\text{blk}}$
  Add (n, k, S, J) to C_{feasible}
end for
return $\arg\min_{(n,k,S,J)\in C_{\text{feasible}}} J$

B.2 最优性证明

定理2（全局最优性）。对于任意有限候选集C和系统状态$\mathbf{s}$，算法1返回在满足可行性约束的条件下使 $J(c)$ 最小化的全局最优配置 c^*。

Proof. Let $C_f = \{c \in C \mid \frac{k_c}{n_c} \geq \alpha \cdot pl\}$ be the feasible set.

1. **可行性**：构造 $\mathbf{y}$，$c \in C_f$，因此C_f满足所有约束。

2. **最优性**：算法1对 C_f进行穷举枚举，计算：

$$c^* = \arg\min_{c\in C_f} J(c)$$

由于 C_f是有限的，因此最小值必然存在，并可通过枚举找到。

3. **全局范围**：任何 $c \notin C_f$都不可行，因此不可能是最优解。因此，c^* 在 C 上是全局最优的。□

B.3 约束满足性证明

定理B.3（解码保证）。在丢包概率为 pl 的独立同分布丢包条件下，TAROT 所选的任何配置都能保证成功解码概率至少为 $1 - \frac{1}{\alpha}$。

证明。设 X为大小为 n的数据块中丢失的源符号数量。在独立同分布丢失条件下：

$$\mathbb{E}[X] = n \cdot pl$$

可行性约束要求：

$$\frac{k}{n} \geq \alpha \cdot pl \Rightarrow k \geq \alpha \cdot n \cdot pl$$

根据马尔可夫不等式：

$$\mathbb{P}(\text{decoding failure}) = \mathbb{P}(X > k) \leq \frac{\mathbb{E}[X]}{k} \leq \frac{n \cdot pl}{\alpha \cdot n \cdot pl} = \frac{1}{\alpha}$$

因此，成功解码的概率满足：

$$\mathbb{P}(\text{success}) \geq 1 - \frac{1}{\alpha}$$

对于典型的 $\alpha = 2$，在期望丢失条件下，这保证解码成功概率至少为 50%。□

B.4 惩罚函数一致性证明

定理 B.4（惩罚函数一致性）。惩罚函数 P_{loss}、P_{over} 和 P_{blk} 满足以下一致性属性：

(1) 单调性：每个惩罚项都会随着条件恶化而增加 *(2)* 凸性：二次和凸惩罚项可确保优化景观平滑 *(3)* 有界性：惩罚项均有界，以防止数值不稳定性 *(4)* 敏感性：适当的缩放可确保各目标之间的影响保持平衡

证明。我们分析每个惩罚函数：

1. 丢失保护惩罚（P_{loss}）：

- **Monotonicity**: $\frac{\partial P_{\text{loss}}}{\partial(n\cdot pl)} = 2\max(0, n \cdot pl - \beta \cdot k) \geq 0$
- **Convexity**: $\frac{\partial^2 P_{\text{loss}}}{\partial(n\cdot pl)^2} = 2 > 0$ (strictly convex)

- **有界性**：上界不受限，但鉴于解码失败的灾难性代价，这是合适的
- **敏感性**：二次形式确保在保护变得至关重要时迅速增大

2. 开销惩罚（P_{over}）：

- **单调性**：$\frac{\partial P_{\text{开销}}}{\partial o} \equiv \alpha_{\text{开销}} \cdot \text{最大值}(0, o - o_{\text{空闲}})^{\alpha_{\text{开销}}-1} \geq 0$
- **凸性**：$\frac{\partial^2 P_{\text{开销}}}{\partial o^2} \geq 0 \quad \alpha \geq 1$ $\alpha =$ 对于开销（当开销时为凸）
- **有界性**：自然受o_{cap}约束限制于σ_{free}
- **敏感性**：凸惩罚抑制过多开销，同时允许必要的保护

3. 分块惩罚（P_{blk}）：

- **单调性**：$\frac{\partial P_{\text{blk}}}{\partial t_{\text{blk}}} = \delta^{-1}$ 对于 $t_{\text{blk}} > \delta(bl)$
- **凸性**：线性惩罚可提供稳定的梯度下降行为

- **有界性**：通过 $\min(1.0, \cdot)$ 运算明确限定上界
- **敏感性**：线性缩放确保对时序违规作出成比例的响应

权重自适应机制确保在任意系统状态$\mathbf{s}$下，主导惩罚项对应于最关键的约束，从而在所有目标之间维持均衡优化。 □

推论 B.5（优化稳定性）。复合目标 $J(c)$ 形成了良好的优化景观，具有以下性质：

(1) 光滑性：所有惩罚项均连续，且几乎处处可微；*(2)* 凸性：可行配置周围存在局部凸区域；*(3)* 数值稳定性：有界梯度可防止优化不稳定；*(4)* 帕累托效率：权重自适应确保对帕累托前沿的探索

B.5 复杂度证明

定理 B.6（时间复杂度）。算法*1*的计算复杂度为 $O(|\mathbf{C}|)$，且常数因子与网络状况无关。

证明。该算法由三个阶段组成：

1. **可行性剪枝**：对每个 $c \in \mathbf{C}$检查 $\frac{k_c}{n_c} \geq \alpha \cdot pl$： $O(|\mathbf{C}|)$次比较。

2. **评分**：对于每个 $c \in \mathbf{C}_f$，计算：

$$P_{\text{loss}}(c) : 4 \text{ operations}$$
$$P_{\text{over}}(c) : 3 \text{ operations}$$
$$P_{\text{blk}}(c) : 6 \text{ operations}$$
$$\text{Weighted sum} : 2 \text{ operations}$$

总计：$15 \cdot |\mathbf{C}_f|$ 次操作。

3. **选择**：在 $|\mathbf{C}_f|$ 个评分中找出最小值： $O(|\mathbf{C}_f|)$次比较。

总体复杂度： $O(|\mathbf{C}|) + O(|\mathbf{C}_f|) = O(|\mathbf{C}|)$，因为 $|\mathbf{C}_f| \leq |\mathbf{C}|$。
使用 $|\mathbf{C}| \approx 400$ 和现代中央处理器，可实现 $\approx 500\mu s$ 执行时间。□

B.6 帕累托最优性证明

定理 B.7（帕累托最优性）。
当所有权重 $w_{\text{loss}}, w_{\text{over}}, w_{\text{blk}} > 0$ 时，该解是帕累托最优的。

证明。反设 c^* 不是帕累托最优的。那么存在 $c' \in \mathbf{C}_f$，使得：

$$P_{\text{loss}}(c') \leq P_{\text{loss}}(c^*)$$
$$P_{\text{over}}(c') \leq P_{\text{over}}(c^*)$$
$$P_{\text{blk}}(c') \leq P_{\text{blk}}(c^*)$$

且至少有一个严格不等式成立。
由于所有权重均为正：

$$w_{\text{loss}} P_{\text{loss}}(c') + w_{\text{over}} P_{\text{over}}(c') + w_{\text{blk}} P_{\text{blk}}(c') < J(c^*)$$

这与 c^* 的最优性相矛盾。因此， c^* 必须是帕累托最优解。□

B.7 单调改进的证明

定理 B.8（单调改进）。对于固定的网络状况，TAROT 选定的配置会随着候选集 C 的扩展而改进（$J(c)$ 减小）。

证明。设 $\mathbf{C}_1 \subseteq \mathbf{C}_2$ 是两个候选集，并设 c_1^*、 c_2^* 是相应的最优解。

由于 $\mathbf{C}_{1f} \subseteq \mathbf{C}_{2f}$（可行子集），我们有：

$$J(c_2^*) = \min_{c \in C_{2f}} J(c) \leq \min_{c \in C_{1f}} J(c) = J(c_1^*)$$

因此，扩展候选集不会使解的质量变差。□

B.8 对测量噪声鲁棒性的证明

定理 B.9（噪声鲁棒性）。设 $\tilde{pl} = pl + \epsilon$为数据包丢包率的带噪估计值，且 $|\epsilon| \leq \epsilon_{\max}$。那么，当 $\alpha \geq \frac{1+\epsilon_{\max}}{1-\epsilon_{\max}/pl}$时，所选配置对于真实丢包率 pl仍然可行。

证明。使用带噪估计值时，可行性约束为：

$$\frac{k}{n} \geq \alpha \cdot \tilde{pl} = \alpha \cdot (pl + \epsilon)$$

为保证鲁棒性，我们要求由此可推出在真实丢包率下仍然可行：

$$\frac{k}{n} \geq \alpha \cdot pl$$

当以下条件成立时，
此结论成立：

$$\alpha \cdot (pl + \epsilon) \geq \alpha \cdot pl \Rightarrow \alpha \cdot \epsilon \geq 0$$

由于 ϵ 可能为负，我们采用最坏情况界：

$$\alpha \cdot (pl - \epsilon_{\max}) \geq \alpha \cdot pl \cdot \delta$$

其中 $\delta < 1$ 表示安全裕度。求解可得关于α的条件。□

参考文献

[1] IETF RFC 6330 (2011). [无日期]. 用于对象传输的 RaptorQ 前向纠错方案。([无日期]). [2] Alhussein A Abouzeid、Sumit Roy 和 Murat Azizoglu。2000。丢失链路上 TCP 的随机建模。载于《IEEE INFOCOM 2000 会议论文集：计算机通信会议。IEEE计算机与通信学会第十九届年度联合会议》（目录编号 00CH37064），第 3 卷。IEEE，1724–1733。[3] Saulius Adamauskas 和 Rytis Krušinskas。2016。移动 5G 技术采用的投资时机：需求波动变化下的决策者聚类与投资意愿。载于《CBU国际会议论文集》，第 4 卷。248–256。[4] Mark Allman、Wesley M Eddy 和 Shawn Ostermann。2003。使用 TCP 估算丢包率。《ACM SIGMETRICS性能评估评论》31，3（2003），12–24。[5] Eitan Altman、Konstantin Avrachenkov 和 Chadi Barakat。2005。具有平稳随机丢包的 TCP/IP 随机模型。《IEEE/ACM网络汇刊》13，2（2005），356–369。[6] Congkai An、Huanhuan Zhang、Shibo Wang、Jingyang Kang、Anfu Zhou、Liang Liu、Huadong Ma、Zili Meng、Delei Ma、Yusheng Dong 等。2025。Tooth：通过云游戏流媒体中的细粒度{前向纠错}，实现视频{体验质量}与{冗余成本}的最佳平衡。载于第 22 届 USENIX网络系统设计与实现研讨会（NSDI 25）。635–651。[7] Abdelhak Bentaleb、Bayan Taani、Ali C Begen、Christian Timmerer 和 Roger Zimmermann。2018。HTTP 流媒体的码率自适应方案综述。《IEEE通信调查与教程》21，1（2018），562–585。[8] Neal Cardwell、Yuchung Cheng、C Stephen Gunn、Soheil Hassas Yeganeh 和 Van Jacobson。2016。BBR：基于拥塞的拥塞控制：测量瓶颈带宽和往返传播时间。《Queue》14，5（2016），20–53。[9] Ke Chen、Han Wang、Shuwen Fang、Xiaotian Li、Minghao Ye 和 H Jonathan Chao。2022。RL-AFEC：基于强化学习的实时视频通信自适应前向纠错。载于第 13 届 ACM多媒体系统会议论文集。96–108。[10] Zhenzhong Chen 和 Hongyi Liu。2014。恰可察觉差异建模：方法与应用。载于《2014 年第 19 届国际数字信号处理会议》。827–830。doi:10.1109/ICDSP.2014.6900782[11] Sheng Cheng、Han Hu、Xinggong Zhang 和 Zongming Guo。2020。DeepRS：用于实时视频通信的基于深度学习的网络自适应前向纠错。载于《2020 年 IEEE国际电路与系统研讨会（ISCAS）》。IEEE，1–5。[12] 消费技术协会。2020。CTA-5004：Web应用视频生态系统——通用媒体客户端数据。[13] Fabio Crugnola。2025。评估和模拟 5G环境下的端到端时延。[14] Sandesh Dhawaskar Sathyanarayana、Kyunghan Lee、Dirk Grunwald 和 Sangtae Ha。2023。Converge：基于体验质量驱动的 WebRTC 多路径视频会议。载于 ACM SIGCOMM 2023 会议论文集。637–653。[15] Kathrin Elmenhorst 和 Nils Aschenbruck。2025。Clouded Comparisons——虚拟机对 TCP-BBR性能的影响。载于 ACM SIGCOMM 2025 海报与演示论文集。127–129。[16] Simone Ferlin、Stepan Kucera、Holger Claussen 和 Özgü Alay。2018。MPTCP 与前向纠错相结合：在异构网络上支持时延敏感型应用。《IEEE/ACM网络汇刊》26，5（2018），2005–2018。[17] Jason Gerard、David C Bonilla、Abdelhak Bentaleb 和 Sandra Céspedes。2024。利用机器学习驱动的自适应前向纠错，优化 WebRTC 的质量与能效。载于 IEEE ICMEW。IEEE，1–6。[18] Richard W Hamming。1950。错误检测与纠正码。《贝尔系统技术期刊》29，2（1950），147–160。[19] Joris Herbots、Mike Vandersanden、Peter Quax 和 Wim Lamotte。2023。Vegvisir：HTTP/3 媒体流媒体测试框架。载于第 14 届 ACM多媒体系统会议论文集。403–409。[20] Joris Herbots、Arno Verstraete、Maarten Wijnants、Peter Quax 和 Wim Lamotte。2023。跨越边界：研究跨层信息共享在增强 QUIC 上自适应码率决策逻辑方面的可行性。载于第 33 届数字音频与视频网络及操作系统支持研讨会论文集。50–57。[21] Han Hu、Sheng Cheng、Xinggong Zhang 和 Zongming Guo。2021。LightFEC：采用轻量级深度学习方法的网络自适应前向纠错。载于第 29 届 ACM多媒体国际会议论文集。3592–3600。[22] J Iyengar 和 I Swett。2021。RFC 9002：QUIC丢包检测与拥塞控制。[23] J Lacan、V Roca、J Peltotalo 和 S Peltotalo。2009。RFC 5510：里德-所罗门前向纠错（前向纠错）方案。[24] TV Lakshman 和 Upamanyu Madhow。2002。具有高带宽时延积和随机丢包的网络中 TCP/IP 的性能。《IEEE/ACM网络汇刊》5，3（2002），336–350。[25] Adam Langley、Alistair Riddoch、Alyssa Wilk、Antonio Vicente、Charles Krasic、Dan Zhang、Fan Yang、Fedor Kouranov、Ian Swett、Janardhan Iyengar 等。2017。QUIC传输协议：设计与互联网规模部署。载于 ACM数据通信特别兴趣小组会议论文集。183–196。[26] Insoo Lee、Seyeon Kim、Sandesh Sathyanarayana、Kyungmin Bin、Song Chong、Kyunghan Lee、Dirk Grunwald 和 Sangtae Ha。2022。R-fec：基于强化学习的前向纠错调整，以提升 WebRTC 的体验质量。载于第30届ACM多媒体国际会议论文集。2948–2956。[27] A Li。2007。RFC 5109：通用前向纠错的 RTP载荷格式。[28] Dumitrel Loghin、Shaofeng Cai、Gang Chen、Tien Tuan Anh Dinh、Feiyi Fan、Qian Lin、Janice Ng、Beng Chin Ooi、Xutao Sun、Quang-Trung Ta 等。2020。5G 对数据驱动技术与应用的冲击。IEEE知识与数据工程汇刊 32，6（2020），1179–1198。[29] Josip Lorincz、Zvonimir Klarin 和 Julije Ožegović。2021。5G网络中的 TCP 拥塞控制综述：研究挑战与未来展望。Sensors 21，13（2021），4510。[30] Francesco Malandrino、Carla-Fabiana Chiasserini 和 Scott Kirkpatrick。2017。面向 5G 的蜂窝网络轨迹：使用、分析与生成。IEEE移动计算汇刊 17，3（2017），529–542。[31] Robin Marx、Maxime Piraux、Peter Quax 和 Wim Lamotte。2020。使用 qlog 和 qvis 调试 QUIC 与 HTTP/3。载于2020年应用网络研究研讨会论文集。58–66。[32] François Michel。2025。coding-for-quic-go：适用于 Go 语言的 QUIC 前向擦除纠错（前向纠错）扩展。https://github.com/francoismichel/coding-for-quic-go。https://github.com/francoismichel/coding-for-quic-go GitHub 仓库。[33] François Michel、Quentin De Coninck 和 Olivier Bonaventure。2019。QUIC-FEC：将前向擦除纠错的优势引入 QUIC。载于2019年 IFIP 网络会议。IEEE，1–9。[34] Mordechai Mushkin 和 Israel Bar-David。1989。吉尔伯特-埃利奥特信道的容量与编码。IEEE信息论汇刊 35，6（1989），1277–1290。[35] Harmony One。2018。go-raptorq：符合 RFC 6330 的 RaptorQ 实现。https://github.com/harmony-one/go-raptorq。https://github.com/harmony-one/go-raptorq GitHub 仓库。[36] P. G. Ortiz。2023。quic-go-fec：适用于 Go 语言的 QUIC 前向擦除纠错扩展。https://github.com/pgOrtiz90/quic-go-fec。https://github.com/pgOrtiz90/quic-go-fec GitHub 仓库。[37] Joachim Otahal。2025。JouMxyzptlk：高分辨率游戏基准测试与全景图。https://joumxyzptlk.de/ 访问日期：2025-04-11。[38] Jitendra Padhye、Victor Firoiu、Donald F Towsley 和 James F Kurose。2002。TCP Reno 性能建模：一个简单模型及其经验验证。IEEE/ACM网络汇刊 8，2（2002），133–145。[39] Mukaddim Pathan、Rajkumar Buyya 和 Athena Vakali。2008。内容分发网络：技术现状、见解与要务。内容分发网络（2008），3–32。[40] Michele Polese、Federico Chiariotti、Elia Bonetto、Filippo Rigotto、Andrea Zanella 和 Michele Zorzi。2019。传输层协议最新进展综述。IEEE通信调查与教程 21，4（2019），3584–3608。[41] Darijo Raca、Dylan Leahy、Cormac J Sreenan 和 Jason J Quinlan。2020。超越吞吐量，迈向下一代：包含信道与上下文指标的 5G数据集。载于第11届ACM多媒体系统会议论文集。303–308。[42] Darijo Raca、Jason J Quinlan、Ahmed H Zahran 和 Cormac J Sreenan。2018。超越吞吐量：包含信道与上下文指标的 4G LTE数据集。载于第9届ACM多媒体系统会议论文集。460–465。[43] Reza Rassool。2017。VMAF 可复现性：验证一种实用的感知视频质量指标。载于2017年 IEEE 宽带多媒体系统与广播国际研讨会（BMSB）。IEEE，1–2。[44] Vincent Roca、Mark Watson 和 Ali C. Begen。2011。前向纠错（前向纠错）框架。RFC 6363。互联网工程任务组（IETF）。https://datatracker.ietf.org/doc/rfc6363/ 提议标准。[45] Jorge Ignacio Sandoval、Jashanjot Singh Sidhu、Sandra Céspedes 和 Abdelhak Bentaleb。2025。5G无线接入网队列管理下的 QUIC 自适应视频流性能。载于2025年 IEEE 地中海通信与网络国际会议（MeditCom）。IEEE，1–6。[46] Sandvine。2024。《全球互联网现象报告》，白皮书。https://www.sandvine.com/global-internet-phenomena-report-2024。技术报告，Sandvine（2024年1月）。访问日期：2024年10月20日。[47] Zaheduzzaman Sarker、Colin Perkins、Varun Singh 和 M Ramalho。2021。用于拥塞控制的 RTP控制协议（RTCP）反馈。互联网 RFC 8888（2021）。[48] Henning Schulzrinne、Steven Casner、R Frederick 和 Van Jacobson。2003。RFC3550：RTP：实时应用的传输协议。[49] Henning Schulzrinne 和 Jonathan Rosenberg。1999。通用前向纠错的 RTP载荷格式。RFC 2733。doi:10.17487/RFC2733[50] Jashanjot Singh Sidhu 和 Abdelhak Bentaleb。2025。CAQ：面向增强视频流媒体的连接感知型自适应QUIC配置。载于第35届数字音频与视频网络及操作系统支持研讨会论文集。8–14。[51] Jashanjot Singh Sidhu 和 Abdelhak Bentaleb。2025。哪些因素能让 QUIC 配置更适合流媒体？载于第4届Mile-High视频会议论文集。97–98。[52] Jashanjot Singh Sidhu、Jorge Ignacio Sandoval、Abdelhak Bentaleb 和 Sandra Céspedes。2025。从5G无线接入网队列动态到播放：性能

分析 QUIC 视频流媒体的性能。*IEEE* 网络学报 34（2025），2384–2399。[53] Kevin Spiteri、Ramesh Sitaraman 和 Daniel Sparacio。2019。从理论到实践：提升 DASH 参考播放器中的比特率自适应。*ACM TOMM* 15，2s（2019），1–29。[54] Jamil Sultan、Mubarak S Mohsen、Nashwan SG Al-Thobhani 和 Waheb A Jabbar。2021。5G 异构网络中的硬切换性能。载于*2021*年新兴智能技术与应用国际会议（*eSmarTA*）。IEEE，1–7。[55] Guibin Tian 和 Yong Liu。2012。迈向动态 HTTP 流媒体中的敏捷平滑视频自适应。载于第*8*届新兴网络实验与技术国际会议论文集。109–120。[56] Jeroen Van Der Hooft、Stefano Petrangeli、Tim Wauters、Rafael Huysegems、Patrice Rondao Alface、Tom Bostoen 和 Filip De Turck。2016。通过 4G/LTE 网络传输 HEVC 视频的基于 HTTP/2 的自适应流媒体。*IEEE*通信

快报 20, 11 (2016), 2177–2180.[57] Tobias Viernickel, Alexander Froemmgen, Amr Rizk, Boris Koldehofe 和 Ralf Steinmetz。2018。多路径 QUIC：一种可部署的多路径传输协议。载于*2018 IEEE* 国际通信会议 *(ICC)*。IEEE，1–7。[58] V Hiroshi Nakayama V Yasuhiro Watanabe 和 V Akihiro Higashi。2008。H. 264/高级视频编码 HDTV 视频编解码器 LSI。*Fujitsu Sci. Tech. J*44, 3 (2008), 351–358. [59] M Watson、T Stockhammer 和 M Luby。2012。RFC 6681：用于 FECFRAME 的 Raptor 前向纠错 (FEC) 方案。[60] Chao Xu、Jessie Hui Wang、Rui Li、Zongpeng Li、Hao Wu 和 Jilong Wang。2024。考虑服务等级协议的低时延传输高效 FEC 方案。载于*NOMS 2024-2024 IEEE* 网络运营与管理研讨会。IEEE，1–9。[61] Feifan Zhang、Ye Li、Jue Wang 和 Tony QS Quek。2021。面向非地面网络的延迟反馈基于学习的 FEC。*IEEE* 通信快报 26, 2 (2021), 306–310.