



清华大学

综合论文训练

面向传输性能提升的跨域 覆盖网络链路优化方法研究

系 别： 电子工程系

专 业： 电子信息科学与技术

姓 名： 高 艺 轩

指导教师： 王 博 副教授

二〇二六年五月

关于论文使用授权的说明

本人完全了解清华大学有关保留、使用综合论文训练论文的规定，即：学校有权保留论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。

作者签名：

导师签名：

日 期：

日 期：

摘 要

跨域覆盖网络需要为用户提供稳定的低延迟、高带宽和低丢包传输服务的同时降低运营成本。传统方案通常依赖专线链路保证服务质量，但专线根据带宽成本计算，且成本较高。已有的成本优化方法主要分为两类，链路调度方法尝试在公网质量较好时使用公网分流，但实际测量表明，公网链路质量下降时段往往与用户流量高峰重合，难以有效削减专线峰值成本。已有端到端冗余编码方法能够缓解丢包影响，但未区分不同链路片段的质量差异，容易在质量良好的片段上引入额外带宽开销。

针对上述问题，本文提出一种面向跨域公网覆盖网络的分段链路质量修复方法。该方法在全部使用公网连接的前提下，仅对低质量链路片段启用前向纠错编码。本文设计了交织 XOR 分组编码方案，以承载各类数据流并降低连续丢包的影响；建立三状态丢包信道模型，根据解码端上报的丢包统计动态选择编码参数；同时设计基于 PI 控制器的输出速率控制机制，缓解 FEC 解码按组恢复导致的突发输出问题。

本文使用 Rust 语言实现了原型系统，并在模拟跨域低质量链路的实验环境中进行验证。实验结果表明，本文方法能够正确识别低质量链路片段并启用 FEC 修复，在无丢包链路片段上保持普通转发。在 0 至 2% 链路丢包率范围内，与直接转发方案相比，本文方法最高达到约 3.6 倍吞吐提升，验证了分段链路质量修复方法的有效性。

关键词：覆盖网络；前向纠错编码；公网链路优化；链路质量修复

Abstract

Cross-domain overlay networks need to provide users with stable low-latency, high-bandwidth, and low-loss transmission services while reducing operational costs. Traditional approaches usually rely on dedicated links to guarantee service quality, but such links are expensive. Existing traffic scheduling methods try to offload traffic to public Internet links when their quality is good. However, real-world measurements show that quality degradation of public Internet links often overlaps with user traffic peaks, making it difficult to effectively reduce the peak bandwidth cost of dedicated links. Existing end-to-end redundancy coding methods can mitigate packet loss, but they do not distinguish quality differences among individual link segments and may introduce unnecessary bandwidth overhead on high-quality segments.

To address these problems, this thesis proposes a segment-level link quality repair method for cross-domain public-Internet overlay networks. Under an all-public-Internet interconnection setting, the method applies forward error correction only to low-quality link segments. It consists of three key components. First, an interleaved XOR block code generates independent repair packets and spreads burst losses across different recovery groups, making the scheme suitable for diverse traffic flows. Second, a three-state packet loss channel model estimates link conditions from decoder-side loss statistics and guides the adaptive selection of coding parameters. Third, a PI-controller-based pacer smooths the bursty packet output introduced by group-based FEC recovery before the packets are delivered to upper-layer protocols.

This thesis implements a prototype system in Rust and evaluates it in an experimental environment that emulates low-quality cross-domain links. The results show that the proposed method correctly identifies low-quality link segments and enables FEC repair on them, while keeping normal forwarding on loss-free segments. With link loss rates from 0 to 2 %, the proposed method achieves up to about 3.6x throughput improvement compared with direct forwarding, demonstrating the effectiveness of segment-level link quality repair.

Keywords: overlay network; forward error correction; public Internet link optimization; link quality repair

目 录

第 1 章 引 言.....	1
1.1 研究背景	1
1.2 研究现状	3
1.3 研究思路与贡献	3
1.4 论文内容	5
第 2 章 背景介绍与研究动机.....	6
2.1 背景介绍	6
2.1.1 覆盖网络	6
2.1.2 网络编码	8
2.2 观察与已有工作不足	8
2.3 研究动机	10
2.4 本章小结	11
第 3 章 相关工作.....	12
3.1 云网络、覆盖网络与隧道技术	12
3.2 网络编码	14
3.2.1 简单复制冗余	15
3.2.2 分组冗余码	16
3.2.3 流式冗余码（Streaming 码）	18
3.3 软件定义网络与网络调度	20
3.4 本章小结	22
第 4 章 跨域云网络传输性能提升研究.....	23
4.1 系统总体架构	23
4.2 交织 XOR 前向纠错编码设计	24
4.3 基于丢包统计的自适应参数调整	26
4.3.1 丢包信道模型	26
4.3.2 参数估计与编码参数搜索	27
4.4 解码端输出速率控制设计	27
4.5 本章小结	29
第 5 章 实验验证与分析.....	30
5.1 实验环境	30

5.2 实验结果与分析	31
第 6 章 结论与展望.....	33
6.1 工作总结	33
6.2 未来工作展望	34
参考文献.....	35
致 谢.....	38
声 明.....	39

插图清单

图 1.1 基于云网络的覆盖网络为用户提供服务.....	2
图 2.1 覆盖网络基于底层网络，将各类物理网络资源抽象为一个虚拟的覆盖网络	7
图 2.2 用户流量高峰与公网链路质量下降时段的重合.....	9
图 2.3 不同公网链路连续七天内平均丢包率热力图.....	10
图 3.1 即使启用了快速重传机制，TCP 仍旧需要一个往返时延才能恢复丢包	14
图 3.2 早期 FEC 工作将冗余信息附加在后续发出的包中进行发送	15
图 3.3 分组码为 n 个数据包附加 k 个冗余包	16
图 3.4 交织编码示意.....	17
图 3.5 分组码需要暂停解码输出等待冗余包到来才能恢复丢包并继续解码过程..	19
图 3.6 混合 SDN 网络	20
图 4.1 系统总体架构.....	24
图 4.2 交织编码矩阵结构示意图 ($d = 4, k = 3$)	26
图 4.3 三状态丢包信道模型.....	27
图 4.4 Pacer 控制模型	28
图 5.1 实验建立的网络拓扑.....	30
图 5.2 不同丢包率下本文方法的吞吐性能提升.....	31

附表清单

符号和缩略语说明

Δh	当前缓冲区深度与目标缓冲区深度的偏差
Δt	相邻两次释放数据包之间的时间间隔
λ	丢包事件的总发生率
d	交织 FEC 编码的交织深度
h	输出速率控制器当前的缓冲区深度
h_0	输出速率控制器目标缓冲区深度
I	输出速率控制器内部维护的积分值
I_{prev}	输出速率控制器上一时刻的积分值
k	交织 FEC 编码中每列保护的数据包数
K_i	PI 控制器的积分系数
K_p	PI 控制器的比例系数
p_{21}	孤立丢包触发概率
p_{23}	连续丢包触发概率
p_{33}	突发延续概率
s	数据包在编码组内的序列号
S_1	网络孤立丢包状态
S_2	网络正常状态
S_3	网络连续丢包状态
v	输出速率控制器计算得到的数据包释放速率
ACK	确认报文 (Acknowledgement)
FEC	前向纠错编码 (Forward Error Correction)
IETF	互联网工程任务组 (Internet Engineering Task Force)
PI 控制器	比例-积分控制器
QoE	用户体验 (Quality of Experience)
QoS	服务质量 (Quality of Service)
RTT	往返时延 (Round Trip Time)
SDN	软件定义网络 (Software Defined Network)
TCP	传输控制协议 (Transmission Control Protocol)

第 1 章 引 言

1.1 研究背景

近年来，实时音视频通信、跨地域文件传输、企业远程办公和云上应用访问等互联网服务快速发展^[1]，网络所承载的业务类型更加多样，它们都需要高质量的网络以维持优秀的用户体验（Quality of Experience, QoE）。例如，在高清实时音视频通讯业务中，参会用户之间存在频繁的实时互动，视频和音频帧必须按时送达，维持稳定的端到端延迟、减少抖动和丢包才能维持优秀的用户体验；相对地，文件下载业务对延迟并不敏感，可用带宽才是影响用户体验的主要因素。尽管这些应用的用户体验评价指标各不相同，但是它们都需要网络提供高吞吐、稳定低延迟且低丢包的高服务质量（Quality of Service, QoS）链路，以维持优秀的用户体验。

然而，随着互联网应用的服务对象从局部区域逐渐扩展到全球范围，新增的大量跨域传输场景中网络状况复杂，使得维持优秀用户体验更加困难。跨国企业协作、跨地域云服务访问、国际在线会议和全球内容分发等场景使得通信双方经常位于不同国家和地区，用户连接不再局限于较短距离的本地网络，而是成为需要跨越多个自治系统、运营商网络和广域互联网链路的跨域连接。用户距离的增加和网络路径的拉长会放大底层网络状态变化对应用体验的影响，例如跨域链路中的拥塞、路由变化和链路质量波动都可能造成延迟升高、带宽下降或丢包增加。端到端优化方法可以在一定程度上缓解网络质量波动，例如通过拥塞控制算法调整发送速率、通过多路径传输绕开部分拥塞路径，但它们无法适配所有跨域网络中可能的情况，在大部分条件下仍旧无法提供优秀的服务质量，进而无法保证优秀用户体验。

为了给各种网络环境下的用户提供一致的高质量服务，网络服务商通常利用覆盖网络（Overlay Network）为跨地域用户建立连接。覆盖网络是一种建立在物理网络之上的逻辑网络，其各部分通常由软件系统统一管理，因而具有部署灵活、扩展方便和易于集中化管理等特点。服务商将底层复杂且动态变化的跨域传输过程隐藏于覆盖网络下，利用覆盖网络灵活可控的转发能力提升用户连接质量。跨域连接的用户需要利用覆盖网络建立连接时，发送端就近通过互联网接入最近的覆盖网络接入网关，数据经由覆盖网络转发至接收端就近接入的覆盖网络网关，经过互联网送达接收端用户，如图 1.1。这种转发方式可以使连接的大部分路径，特别是跨域传输阶段，由服务商可管理的覆盖网络承载，服务商也就能够通过优化覆盖网络内的转发节点、转发路径和链路质量来提升用户体验。

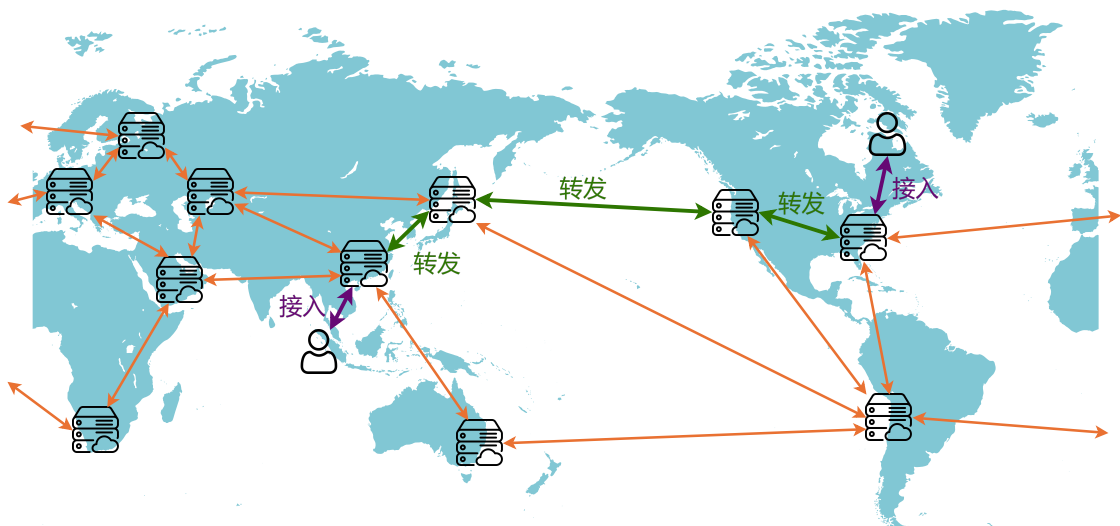


图 1.1 基于云网络的覆盖网络为用户提供服务

当今的覆盖网络通常构建在云网络资源之上，云网络服务商提供多种资源供服务商按需选择。随着云计算和跨地域互联网服务的发展，服务商在构建覆盖网络时通常不再自行建设底层基础设施，而是租用云服务商在全球多个地区提供的服务器、网关和链路等计算与网络资源，并将这些资源抽象为覆盖网络中的转发节点、接入网关和逻辑链路。由于这些资源可以通过软件配置进行申请、扩展和调整，服务商能够根据业务覆盖范围、用户分布和流量需求快速构建跨地域覆盖网络。例如，当服务商需要连接不同地区的数据中心或接入点时，可以在相应区域租用计算节点作为转发网关，并在节点之间选择合适的网络链路，从而形成一条面向用户连接的覆盖网络转发路径。因此，覆盖网络的灵活性在很大程度上来自云网络资源的可配置性，而覆盖网络的服务质量也受到所选云资源质量的直接影响。

不同云资源的质量与定价有所区别，维持高网络服务质量同时降低成本是当今研究的重点。覆盖网络中同一条逻辑链路的连接通常可以由多条物理链路抽象而成，云网络服务商往往同时提供专线链路和公网链路等不同选择。专线链路通常延迟、丢包率都较低且稳定，能够提供较好的传输质量和用户体验，但其价格较高，且常按照流量峰值计费，大规模使用会给服务商带来较高的运营成本；公网链路价格较低，计费方式也更灵活，但容易受到其他用户流量的影响，在拥塞和竞争下出现延迟升高、丢包增加和带宽波动等问题，质量不稳定。全部使用专线链路可以较好地满足业务服务质量需求，但成本难以控制；完全依赖公网链路又可能导致服务质量无法稳定满足用户体验要求。因此，如何在保证覆盖网络高服务质量的同时尽可能降低资源使用成本，成为覆盖网络优化中亟须解决的问题。

1.2 研究现状

现有的研究方案主要从两类视角审视覆盖网络传输，并对其进行优化。一些工作从覆盖网络的角度对覆盖网络的调度、链路选择等进行优化；另外一些工作从端到端的角度进行优化，对发送速率、发送内容等进行调控和编码。

链路调度类的工作从覆盖网络管理者的角度出发，通过网络路径选择等手段，利用低成本链路质量较好的时间窗口分担流量。这些工作在对连接两端用户透明的前提下，利用覆盖网络中同一链路可由质量价格不同的多个链路抽象而来的特点，通过不断监控同一逻辑链路下的公网链路与专线链路的质量，并在公网质量优秀可以为用户提供优质服务的时段将部分流量经由公网链路发送。这些工作利用公网上分担部分流量，降低了在专线上发送的数据流量，从而降低使用专线的成本^[2,3]。

冗余编码类的工作从端到端用户的角度出发，在对转发覆盖网络透明的前提下，将发送端的发送策略随网络进行调整，以提升传输性能。一些工作通过主动探测网络状态调整发送速率，以充分利用网络带宽^[4-6]。另外一些工作通过在发送端设计特殊的网络编码以应对传输过程中可能的丢包。这些工作大多应用前向纠错编码等编码，通过发送额外的冗余信息，使接收端从冗余信息恢复丢失包，从而降低上层应用感知到的丢包，提升了用户感知到的链路质量^[7-9]。

然而，这两类方法在跨域公网链路场景下仍存在局限。链路调度类工作希望在公网链路质量较好时利用其分担部分流量，但这种方法依赖低成本链路在业务高峰时段仍能提供足够稳定的服务质量；而在实际网络中，用户需求较高的时段往往也伴随着公网链路拥塞和质量下降，导致公网链路难以承载大量高质量传输，专线峰值流量仍然难以有效降低。端到端优化类工作则将整条传输路径视为不可区分的整体，只能根据端到端观测结果调整发送速率或加入冗余信息，难以判断性能瓶颈究竟来自路径中的哪一段链路。为了应对可能发生的丢包，这类方法通常需要对整条路径上的流量统一加入冗余，即使部分链路片段质量良好也会产生额外带宽开销；同时，单纯降低发送速率虽然可以缓解拥塞，却无法直接修复低质量链路上的丢包问题。上述局限表明，现有方法要么从覆盖网络层面规避低质量公网链路，要么从端到端层面被动适应路径质量变化，尚未充分利用覆盖网络中路径可拆分、链路可感知和节点可控的特点。

1.3 研究思路与贡献

本文的核心观察是覆盖网络中的不同公网链路片段的性质差异大，应分段进行优化。覆盖网络中，部分跨域链路由于竞争激烈、延迟高，导致性能低下，而部

分域内链路性能优秀，与专线质量接近，已有的工作没有考虑到覆盖网络中不同链路片段的性质差异，导致优化效果不佳。本文提出应该站在链路层级上，对不同质量的链路分别进行针对性的传输优化。为实现对网络中不同链路的针对性质量提升，本文需要解决以下两个挑战：

1. **如何应对各个链路片段频繁且不可预测的链路质量变化。**公网链路的丢包率和连续丢包模式会随时间变化，若长期对所有链路使用固定冗余，会带来不必要的带宽开销；若冗余不足，又无法有效修复低质量链路。因此，系统需要根据实时链路状态判断是否启用冗余，并动态选择合适的编码参数。
2. **如何在对传输两端透明的前提下提升部分链路的传输质量。**覆盖网络承载的上层业务类型多样，传输两端通常并不知道中间覆盖网络的转发细节，也难以配合覆盖网络内部的链路优化。因此，优化机制不能依赖修改应用报文、改变端到端协议语义或要求发送端与接收端协同，而需要能够在覆盖网络内部的单个低质量链路片段上独立完成质量修复。同时，链路片段级修复还必须保持端到端数据包的正常传输节奏，避免因中间节点处理造成突发交付、乱序或额外时延，从而干扰上层拥塞控制和实时应用体验。

基于此，本文设计了一套基于交织前向纠错编码（Interleaved Forward Error Correction, Interleaved FEC）的跨域公网链路优化方法。本文提出的方法使用公网实现覆盖网络中所有节点的互联，对覆盖网络中的每一段链路，通过监控链路上的丢包情况，利用马尔科夫链建模网络丢包模型，对低质量的链路动态选择 FEC 编码参数，并利用交织 XOR 编码进行编码和丢包恢复，并在解码时对输出速率利用比例-积分控制器进行动态平滑处理。本方法不需要使用专线连接，极大地降低了链路的使用成本，同时又有选择性地在低质量链路上使用冗余编码，避免了在高质量链路上添加额外带宽。另外，应用交织编码技术，将冗余包与数据包间隔其它数据包发送，极大地降低了链路连续丢包对丢包恢复的影响。

本文实现了基于本文提出的分段链路质量优化方法的分布式覆盖网络转发以及针对低质量链路的冗余包计算及丢包恢复算法。经过对真实网络的模拟实验，本文提出的方法相比直接使用公网链路将端到端带宽提升了最高 260 %。

总结而言，本文主要的贡献是：

- 通过对公网链路的真实测量，指出了长距离跨域公网链路质量差的核心在于不同公网链路质量差距大、部分跨域链路片段存在链路质量差的特性；
- 提出了通过针对性地对低质量链路片段加入冗余，以最低的额外带宽开销实现对链路整体质量的提升；
- 实现并测量了本文提出的链路优化方法在跨域公网链路场景下对端到端性

能的提升。

1.4 论文内容

本文共分为六章，各章内容安排如下：

第 1 章为引言。本章介绍跨域云网络中覆盖网络传输的应用背景，分析专线链路成本高、公网链路质量不稳定所带来的矛盾，概述现有链路调度与冗余编码方法的不足，并给出本文的研究思路与主要贡献。

第 2 章为背景介绍与研究动机。本章介绍云网络、覆盖网络以及前向纠错编码等必要背景，结合真实测量结果分析现有方法在跨域公网链路场景下的局限，指出低质量公网链路不应仅被规避，而应结合覆盖网络的分段转发能力进行针对性质量修复。

第 3 章为相关工作。本章分别介绍覆盖网络与隧道技术、链路质量优化方法以及软件定义网络与网络调度相关研究，并分析这些工作与本文研究问题之间的联系和差异。

第 4 章为跨域云网络传输性能提升方法。本章介绍本文提出的全公网链路优化系统，包括系统总体架构、交织 XOR 前向纠错编码设计、基于丢包统计的自适应参数调整方法，以及用于平滑解码端突发输出的速率控制机制。

第 5 章为实验验证与分析。本章介绍实验环境与实验设置，通过在模拟低质量链路条件下对比直接转发方案和本文方法的端到端吞吐性能，验证本文提出的分段链路质量修复方法的有效性。

第 6 章为结论与展望。本章总结全文的主要工作与实验结论，并讨论本文方法仍存在的不足以及未来可进一步优化的方向。

第 2 章 背景介绍与研究动机

本章首先在 2.1 节中介绍覆盖网络和网络编码的基本信息，介绍在维持高服务质量前提下进行成本优化这一核心问题。之后在 2.2 节介绍已有工作在实际场景下的不足。最后，在 2.3 节介绍本文提出的通过对云网络公网进行分段质量修复的基本思路及面临的挑战。

2.1 背景介绍

2.1.1 覆盖网络

覆盖网络（Overlay Network）是一种已经获得广泛应用的网络虚拟化设计，它是基于物理的底层网络（Underlay Network）上通过对资源的逻辑整合而形成的逻辑网络。如图 2.1，覆盖网络在已有的硬件网络上构建一个虚拟的网络层，使得使用覆盖网络服务的企业和用户可以获得更灵活与稳定的虚拟网络连接。近年来，企业对虚拟化和云网络的需求不断增长，因而将分布在全球各地的云资源进行互联的需求也不断提升。企业构建覆盖网络以承载业务时，通常不选择自己搭建基础设施，而是选择向云服务商按需租用计算和网络资源，并利用这些云网络资源组成覆盖网络。

从用户和业务方的角度看，覆盖网络的价值不只在于完成两端连通，还在于屏蔽底层网络路径、运营商和云区域差异带来的复杂性。用户只需要接入邻近的覆盖网络网关，后续路径选择、故障绕行、链路切换和质量优化由覆盖网络统一完成，因而能够在不直接管理底层网络资源的情况下获得更可控的跨地域传输服务。

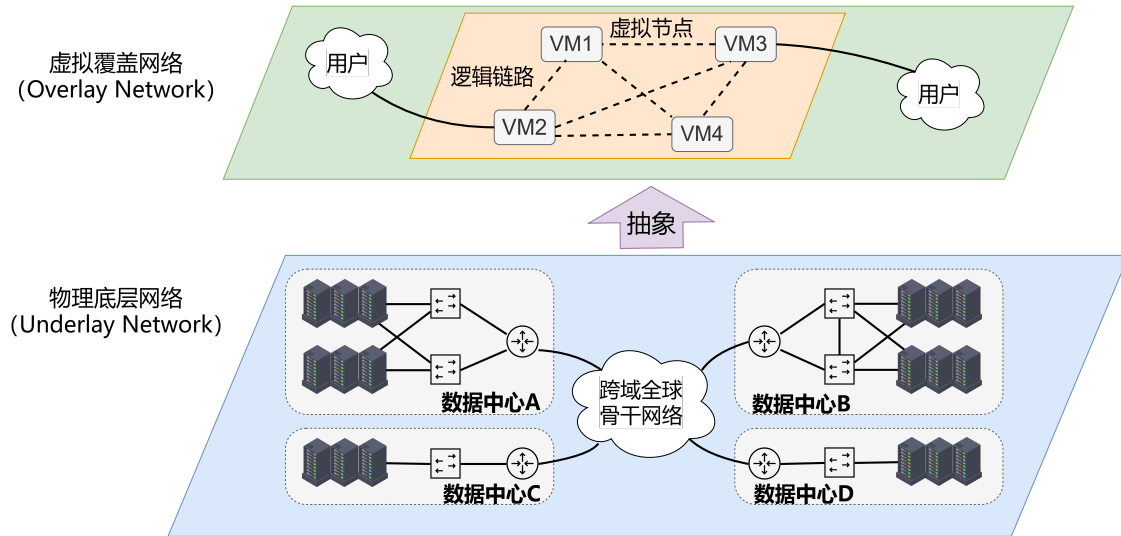


图 2.1 覆盖网络基于底层网络，将各类物理网络资源抽象为一个虚拟的覆盖网络

在覆盖网络中，不同节点之间通常存在多条物理连接路径，这些物理链路具有不同的链路质量和不同的链路价格及计费方式。例如，许多云服务商同时提供公网链路互联与专线链路互联^[10,11]。其中，专线链路通常具有更稳定的传输性能、更低的丢包率与时延，但部署成本与使用成本较高；而公网链路虽然成本较低，却容易受到网络拥塞、跨域路由波动等因素影响，出现高丢包、时延抖动等问题^[2]。与此同时，链路质量与网络负载往往还会随着时间动态变化，使得不同链路在不同时间段内呈现出不同的性能特征。

随着云计算与实时互联网应用的发展，现代云网络中的跨域流量规模持续增长，用户对于传输质量与服务稳定性的要求也不断提高。传统的覆盖网络服务商为了为用户提供高质量的传输服务，确保能为用户持续稳定提供低延迟、高带宽、低丢包的转发路径，选择尽可能多地使用专线链路构建覆盖网络，而这对运营成本带来了较大的压力。如何在维持网络服务质量保持高带宽、低丢包、低延迟的前提下，尽可能减少构建和运营云网络所需的成本，是各服务商关注的重点。

一些工作意识到了公网链路与专线链路在经常存在定价差异，因而尝试在维持覆盖网络服务质量的前提下，利用覆盖网络易于实时配置的特性，将部分流量转移至质量优秀的公网链路上，以减少专线链路的压力^[2,3]。这些工作在公网链路质量较好时，利用低价的公网链路为部分用户提供服务，降低了高价专线需要承载的流量，从而在服务流量总量不变的情况下，降低了高价流量的占比，进而降低了链路部署的总成本。

2.1.2 网络编码

公网链路质量下降最直接的表现之一是数据包丢失。对于可靠传输协议而言，丢包通常需要依赖重传机制恢复；然而在跨地域云网络中，端到端往返时延较高，重传一次丢失的数据包往往需要等待一个完整的往返时延，容易造成吞吐下降和实时业务卡顿。因此，在低质量链路上仅依赖端到端重传机制，难以满足实时音视频、交互式应用等业务对低延迟和稳定性的需求。

为了应对网络中的丢包，一些工作针对不同的应用类型和需求设计了不同的网络编码承载应用数据。前向纠错编码（Forward Error Correction, FEC）是一类常见的用于优化链路质量的网络编码，其基本思想是在发送原始数据的同时加入一定数量的冗余信息，使接收端在部分数据包丢失时，可以利用已经收到的数据包和冗余包直接恢复丢失内容，而不必等待发送端重传。与重传机制相比，FEC 通过额外带宽开销换取更短的丢包恢复时间，因而适合用于对时延敏感、但又需要在不稳定网络上持续传输的场景。

常见的 FEC 方案包括简单复制、XOR 码、Reed-Solomon 码^[12]以及流式编码^[13]等。它们在冗余效率、计算开销、连续丢包恢复能力和恢复延迟方面各有侧重。例如，分组码将多个数据包组织为一个编码组，并为该组生成独立的冗余包，能够以较低的实现复杂度恢复一定数量的丢包；交织技术则通过改变数据包与冗余包的组织和发送顺序，将连续突发丢包分散到不同的恢复单元中，从而降低单个编码组内同时丢失多个数据包的概率。这些技术为在不依赖重传的情况下改善低质量公网链路的传输质量提供了基础。

2.2 观察与已有工作不足

1. 公网链路质量下降与用户流量高峰重合，基于公网分流的调度方法难以削减专线峰值成本。

已有的链路调度类工作没有考虑到公网链路质量下降的时间段与用户流量高峰有明显的相关性，公网链路的实际分流能力有限。如图 2.2，在某企业的某条公网连接中，用户流量带宽提升的时段与丢包率提升、延迟波动的时段有较强的相关性，只在公网丢包低、延迟稳定的时段使用公网链路只能削减专线上承载的一小部分流量。进一步地，由于用户流量带宽较大的时段公网持续恶化，这些方法也不能利用公网链路削减专线需要承载的峰值带宽，使得专线链路仍然在传输流量时起主导作用，链路使用成本的削减程度有限。另外，与公网链路通常可以灵活选用按量付费与按峰值带宽付费不同，专线链路通常只能按一段时间内的峰值带宽或 95 分位带宽付费^[14,15]，这些方法不能有效地削减专线上承载的峰值带宽

就意味着专线的使用成本不会由于公网的部分分流而显著降低。因此，这些工作对链路使用成本的削减十分有限，甚至可能由于专线链路成本没有明显下降，反而由于额外使用公网链路而导致链路使用成本增加。

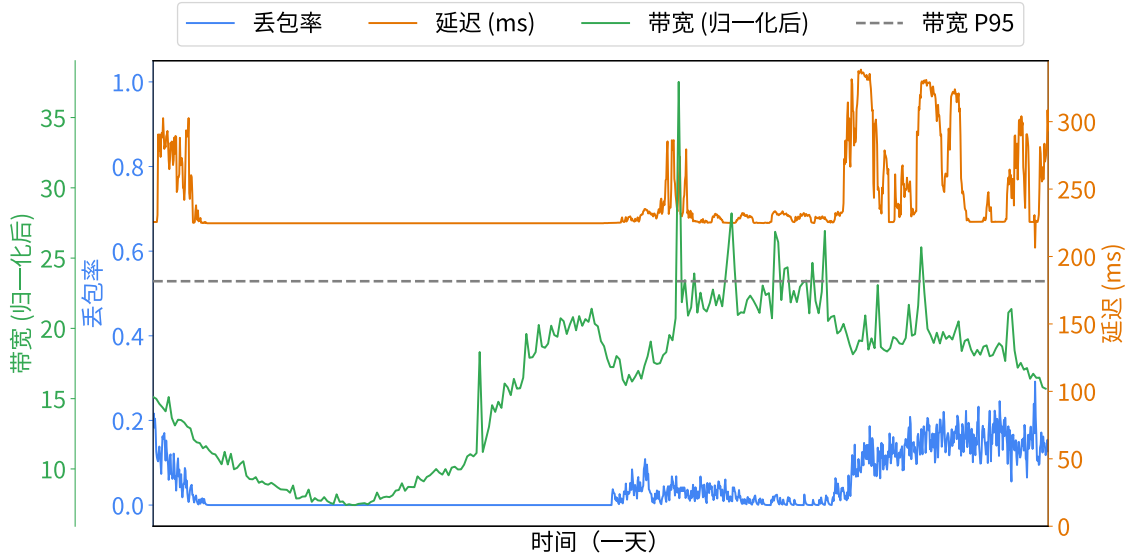


图 2.2 用户流量高峰与公网链路质量下降时段的重合

2. 公网链路不同分段质量差异显著，端到端进行网络编码带宽浪费严重。

云服务商只对专线链路的质量提供服务质量保证（Service level agreement, SLA），而对公网的具体性能没有任何形式的保证。尽管服务商不对公网的性能做出任何保证，所有的公网链路在所有的时间段质量都劣于专线链路。如图 2.3 所示，部分公网链路有着较低的平均丢包率，质量几乎与专线相当，而只有部分链路，特别是跨域链路的丢包率较高，链路质量较差，与低丢包的专线有较大差距。覆盖网络对用户流量进行转发时，通常将多个不同的网络片段相连组成连接两侧接入网关的路径。由于覆盖网络的内部转发机制通常对端到端的传输透明，两端的客户端只能感知到由多个链路的丢包级联而成的最终丢包率，只要组成转发路径的链路中包含了至少一条丢包率较高的跨域公网链路，端到端感知到的丢包率就会明显上升。这导致在跨域连接的场景下，网络编码类工作只能以感知到的高丢包率对在整个路径上转发的包加入大量的冗余，造成了较大的带宽浪费，造成链路使用成本上升。例如，一条端到端的连接可能分别由 AB、BC、CD 三段链路组成，其中三段链路的丢包率分别为 0, 0.2, 0，如果使用端到端的冗余，为了应对其中一条高丢包链路带来的丢包，必须在整条链路上加入额外约 20% 的冗余，导致在从 A 发往 D 的流量中，AB、BC 两段都承载了相比有效数据多 20% 的数据量。在这种场景下，AB 段网络本身质量优秀，但是却由于后段质量较差的网络额外承载了流量，导致了链路使用成本上升。

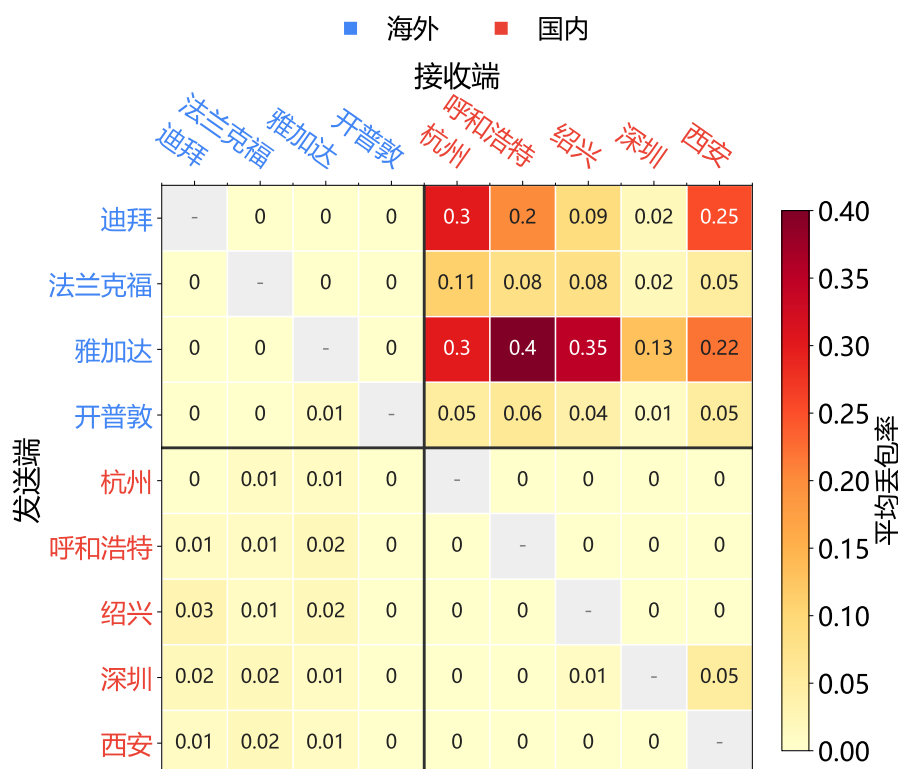


图 2.3 不同公网链路连续七天内平均丢包率热力图

2.3 研究动机

前文的观察表明，现有方法在跨域云网络场景下仍然存在局限。一方面，链路调度类方法主要利用公网质量较好时的机会窗口，将部分流量从专线迁移到公网；但当用户流量进入高峰期时，公网链路也更容易出现丢包和抖动，系统仍然需要依赖专线承载主要流量，因而难以真正降低按峰值带宽计费的专线成本。另一方面，网络编码类方法虽然能够修复丢包，但通常将端到端路径视为一条整体链路，在整条路径上统一添加冗余，没有区分不同链路片段之间的质量差异，容易在质量良好的片段上引入不必要的带宽开销。

因此，本文的基本思路是：不再将低质量公网链路视为只能由调度算法规避的不可用资源，而是利用覆盖网络中间节点可控、路径可分段的特点，对公网转发路径进行链路粒度的质量修复。具体而言，系统完全基于成本较低的公网链路构建覆盖网络，并持续监控各个链路片段的传输质量；对于质量良好的片段，系统保持普通转发，避免引入额外开销；对于丢包率较高或存在连续突发丢包的低质量片段，系统在该片段两端加入前向纠错编码，将质量修复限制在真正发生问题的链路范围内。通过这种方式，本文希望在不依赖专线链路的条件下，使全公网覆盖网络在用户高需求时段仍能提供接近专线链路的传输质量，同时避免端到端统

一冗余带来的带宽浪费。

围绕这一思路，系统设计需要进一步解决以下三个核心挑战。

1. **如何在通用覆盖网络转发路径中透明地加入片段级冗余编码。**覆盖网络承载的上层业务类型多样，用户数据包大小和发送节奏并不固定，部分数据包可能已经接近最大传输单元。因此，编码机制不能依赖修改用户报文内容或在用户包内部预留冗余空间，而需要以独立、透明的方式插入相邻覆盖网络节点之间的链路片段，并有效应对该片段上的连续丢包。
2. **如何根据链路质量变化自适应选择冗余强度。**公网链路的丢包率和突发丢包模式会随时间变化，固定冗余参数难以同时适应不同链路和不同时间段的网络状态。冗余不足会降低丢包恢复能力，冗余过高则会消耗额外带宽并增加解码等待时间。因此，系统需要根据实时链路观测动态决定是否启用 FEC 以及相应的冗余保护强度。
3. **如何避免片段级丢包恢复影响端到端传输节奏。**FEC 解码通常以编码组为单位恢复数据包，恢复完成后可能在短时间内集中交付多个数据包。这种突发式输出会改变接收端观测到的数据到达节奏，并进一步影响拥塞控制算法的速率估计和实时应用的播放稳定性。因此，系统在修复丢包的同时，还需要对解码后的输出过程进行平滑控制。

2.4 本章小结

本章围绕跨域云网络中的传输质量与链路成本问题，介绍了云网络、覆盖网络以及前向纠错编码等背景知识。首先，覆盖网络为跨地域云资源互联提供了灵活的虚拟网络抽象，但公网链路质量不稳定、专线链路成本较高，使得服务商需要在传输质量和运营成本之间进行权衡。其次，FEC 等网络编码技术能够通过冗余信息恢复部分丢包，为改善低质量公网链路提供了基础。

在此基础上，本章结合真实网络测量指出现有方法的两点不足：一方面，公网链路质量下降时段与用户流量高峰存在明显重合，使得仅依赖公网分流的链路调度方法难以削减专线峰值成本；另一方面，不同公网链路片段之间质量差异显著，端到端统一添加冗余会在高质量片段上造成额外带宽浪费。基于这些观察，本文提出利用覆盖网络路径可分段、中间节点可控的特点，在全公网互联的前提下对低质量链路片段进行针对性质量修复，并进一步总结了该思路在编码透明性、参数自适应和解码输出平滑等方面面临的设计挑战。

第 3 章 相关工作

本章主要介绍一些与本文工作相关的已有工作。3.1 节主要介绍了当前在工业界得到较为广泛使用的用于构建覆盖网络的几种隧道技术，3.2 节介绍网络编码的基本概念，并重点介绍多种常用于链路丢包恢复的前向纠错编码技术，比较它们的异同。最后，3.3 节介绍了与覆盖网络紧密联系的软件定义网络（Software Defined Network, SDN）的概念，以及与其相关的一些工作。

3.1 云网络、覆盖网络与隧道技术

云网络是服务商将计算和网络基础设施作为一种服务进行售卖（Infrastructure as a Service, IaaS）的新型计算范式^[16]。它的核心思想是云网络的服务商出资搭建数据中心、购买网络资源将数据中心内的计算、存储等单元连接互联网并将这些资源出租，其他服务提供商或者个人用户可按需要购买云服务商中提供的资源，并通过互联网访问。与传统的网络依赖与本地硬件进行部署不同，云网络通过虚拟机、虚拟路由器、虚拟交换机、负载均衡、虚拟防火墙等多种技术将已有的物理网络和计算资源抽象为虚拟化的计算资源，提供给不同的用户进行访问。通过网络虚拟化技术，云网络同时减少了计算资源的提供商与用户的成本，云网络的虚拟化特性使得资源可以按需用户需求动态分配与计费，用户只需为自己真正使用的资源付费，同时云服务商可以通过对虚拟资源在硬件上的整合避免资源分配后的浪费，高效地满足所有用户的资源需求，降低运营成本^[17]。

在上述云网络资源基础上构建覆盖网络，已经成为当前企业与网络服务商进行跨地域业务部署的常见运营策略。对于需要连接多地用户、数据中心或云上服务的企业而言，直接自建全球范围内的物理网络基础设施不仅部署周期长，而且需要承担较高的建设和维护成本。相比之下，企业可以按需租用云服务商在不同地域提供的虚拟机、网关以及公网或专线链路，将这些分散的云网络资源抽象为覆盖网络中的转发节点与逻辑链路，并通过软件方式完成路径配置、节点扩展和链路调整。由此，覆盖网络既继承了云网络按需部署和弹性扩展的优势，也为后续根据链路质量与成本进行灵活调度提供了基础。然而，要在已有的底层 IP 网络之上形成彼此隔离、可配置且对上层业务透明的逻辑网络，仅依赖云资源本身并不足够，还需要相应的封装与转发机制。

覆盖网络的实现依赖于隧道封装技术，其基本原理是将原始的二层或三层报文封装在另一种网络协议中进行传输，从而在底层的 IP 网络上构建虚拟的二层网络。

当前主流的 Overlay 隧道技术主要包括 VXLAN^[18]、NVGRE^[19]和 Geneve^[20]等，它们在封装格式、协议机制和适用场景上各有特点。

VXLAN (Virtual eXtensible Local Area Network, 虚拟可扩展局域网)^[18]是由 IETF 制定的虚拟网络技术之一，广泛应用于在数据中心和云网络中。VXLAN 通过 MAC over UDP 的方式，将二层的以太网帧封装在 UDP 报文中通过公网传递，对虚拟的二层网络在三层网络的基础上进行扩展。VXLAN 使用 24 比特的虚拟网络标识 (VXLAN Network ID, VNI) 来区分不同的虚拟以太网，可以突破传统 VLAN 的 4096 个虚拟网络数量限制，提供约 1600 万个各自独立的虚拟局域网。VXLAN 协议将普通的二层网络数据帧添加上 VXLAN 的包头，之后再将数据包装上外层的以太网、IP 和 UDP 报文头后发送至公网。VXLAN 包的封装和解封装由 VXLAN 隧道端点 (VXLAN Tunnel End Point) 进行，VTEP 负责将从虚拟机进入隧道的包进行封装，也负责将从隧道接收到的包进行解封装后交付给虚拟机。这使得 VXLAN 隧道对虚拟机透明，便于与其他网络系统集成。VXLAN 利用已有的 UDP 传输机制在网络中建立隧道，成熟度高，当前已广泛应用于数据中心。

NVGRE (Network Virtualization using Generic Routing Encapsulation, 基于路由封装的网络虚拟化)^[19]是另一种主要的虚拟隧道协议。该协议主要应用于微软的 Hyper-V 虚拟环境中^[21]。NVGRE 将二层的 MAC 包封装在 GRE 隧道包内通过公网传递，利用 GRE 协议中的 Key 字段传递包所属的虚拟子网标识 (Virtual Subnet ID, VSID) 以及流标识 (FlowID)。NVGRE 同样以 24 比特标识虚拟网络的名称，因此也可以支持最多约 1600 万个虚拟子网。同时，NVGRE 支持在同一子网内进一步通过流标识来区分不同的数据流，为更精细地管理流量和流量均衡提供了支撑。然而，这要求物理网络设备具备识别和处理这些字段的能力，对在公网部署带来了一定的挑战。

Geneve (Generic Network Virtualization Encapsulation, 通用虚拟化网络封装技术)^[20]是 IETF 新提出的通用网络虚拟化封装协议，旨在以单一、可扩展的封装格式取代碎片化的 VXLAN、NVGRE 等多种隧道协议，以维持生态统一。Geneve 也采用 MAC over UDP 的封装，通过灵活配置的元数据传递机制满足多种网络虚拟化需求。Geneve 也使用 24 比特的虚拟网络标识 (Virtual Network Identifier, VNI) 来区分不同的虚拟网络，支持的网络数量与 VXLAN、NVGRE 等协议相当。与 VXLAN 等协议不同，Geneve 允许在头部后添加可变长度和数量的控制位和控制信息，可以有效满足不同虚拟网络的需求，增强了可扩展性。Geneve 协议通过设计可选的元数据空间，允许在不修改协议的前提下引入新功能，自推出以来已经逐步得到各类虚拟网络平台的支持^[22,23]，但是协议较为复杂，适配难度较大。

3.2 网络编码

低质量的互联网链路由于负载较大出现拥塞或部分设备运行故障时，容易出现丢包或者延迟波动。在这些低质量的链路上进行传输时，即使链路还有可用的传输带宽，也会出现丢包或是延迟波动。即使 TCP^[24] 等可靠传输协议通过重传确保了所有数据都能可到达，但性能较差。这是因为 TCP 协议依靠超时重传来在确保所有数据都最终送达至接收端，即使使用了基于重复 ACK 的快速重传机制，如图 3.1，恢复单个丢失的包也至少要经历接收端检测丢包——请求发送端重传——发送端重传包送达恢复的过程，至少需要一个往返时延（Round Trip Time, RTT）才能恢复。对于一条在云网络中的跨域链路，往返时延可能达到 300 ms 或更长，如此缓慢的丢包恢复不仅会阻塞后续数据包的发送，也会极大地影响实时媒体服务如影视直播、视频通话等应用的用户体验。

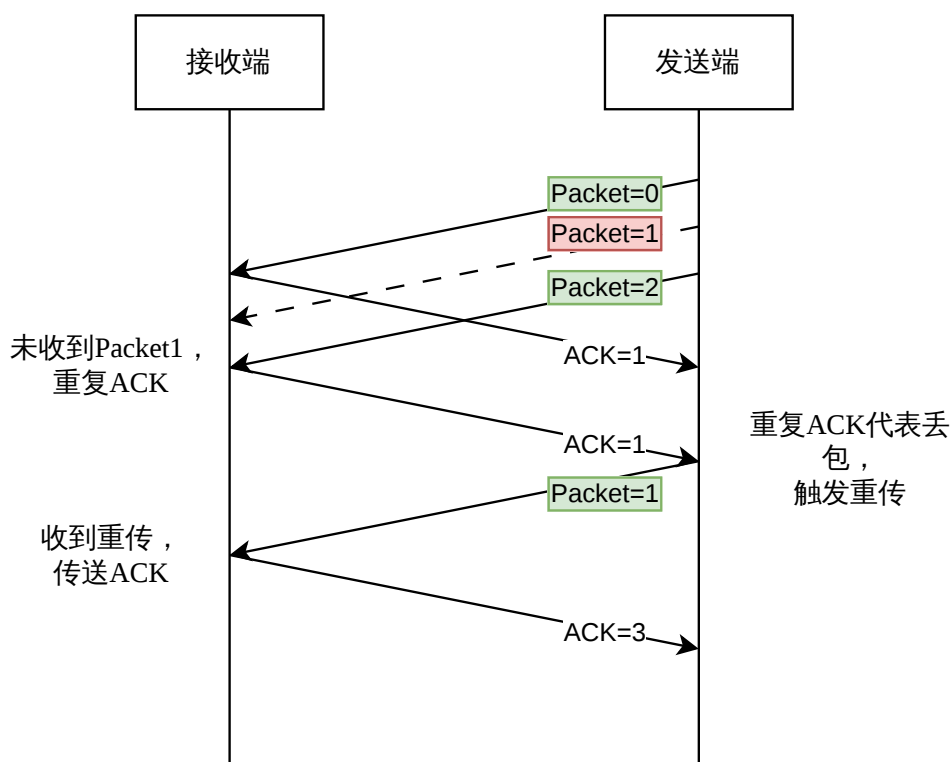


图 3.1 即使启用了快速重传机制，TCP 仍旧需要一个往返时延才能恢复丢包

针对此问题，研究者们提出了多种解决方案。网络编码（Network Coding）是一类在数据传输过程中对多个数据单元进行组合、冗余生成或重新编码的技术，其核心思想是不再只按原样转发单个数据包，而是通过编码数据包携带多个原始数据包之间的关联信息，使接收端能够在收到足够信息后恢复原始数据。网络编码

可以用于提高吞吐、增强可靠性或改善链路资源利用率；在链路丢包修复场景中，最常用的一类网络编码方法是前向纠错编码（Forward Error Correction, FEC）。

FEC 的基本思想是，在发送数据时直接加入一部分冗余信息，以确保在部分信息丢失时，接收端无需请求发送端重新传送任何信息，而可以利用已经接收到的信息配合冗余信息推算出丢失的信息。与重传机制应对丢包需要经历一整个往返时延的长时间反馈路径不同，利用前向纠错编码的冗余包进行恢复只需要等待后续冗余包送达后即可进行，错误恢复时间短，能更好地适应延迟敏感型应用如视频通话等应用的需求。

为了实现高效的前向纠错编码，研究者们提出了多种编码方式，它们针对不同的目标进行了设计和优化。

3.2.1 简单复制冗余

早期的 FEC 工作主要通过对时间敏感的数据包进行简单地复制和多次传输进行错误恢复。如图 3.2 所示，通过将每个数据包的内容复制多份，每次发送新的数据的同时，在同一个数据包中同时捎带发送之前已经发送过的一些数据包，这样可以在一部分数据包丢失的同时仍旧确保接收端收到了所有数据。Bolot 等人^[7]基于此思路提出可以利用实时语音通话应用中已经存在的平均丢包率监控字段对传输链路的丢包模式和丢包律进行估计，从而动态地选择重复发送包的发送间隔和次数，优化通话用户的用户体验。之后 Gandikota 等人^[25]在此基础上提出可以通过多路径传输，进一步提升冗余包和原始数据包中至少有一个送达的概率。Gandikota 等人在工作中提出通过估算网络中的丢包率，动态地调整冗余参数以实现对语音流中的重要子流进行保护，同时再将编码后的数据包以及其他次要子流经过两个最大程度节点不相交路径在网络上与重要数据流分开传输，以降低数据传输丢失概率、提升用户体验。Huang 等人^[8]通过测量 Skype 应用在不同丢包网络条件下的行为，印证了相关冗余编码在提升实时语音通话用户体验方面的积极作用。

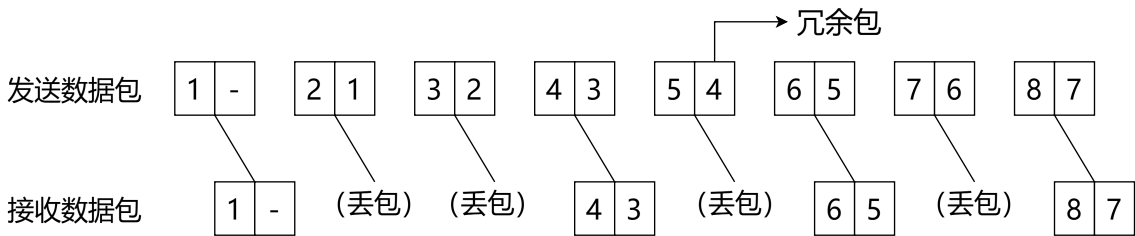


图 3.2 早期 FEC 工作将冗余信息附加在后续发出的包中进行发送

3.2.2 分组冗余码

通过重复发送数据包的方式添加冗余虽然简单，但是会带来较高的冗余开销，为了提高冗余信息的恢复效率，研究者们进一步提出了基于分组冗余码的前向纠错机制。XOR 码和 R-S 码是较为主要的冗余纠错恢复机制。如图 3.3，这两种编码都是线性分组码，将原始数据分为 n 个数据包一组，对于每一组数据再加入 k 个冗余数据包并将 $n + k$ 个数据一并发送，接收端同样以组为单位进行丢包的恢复。

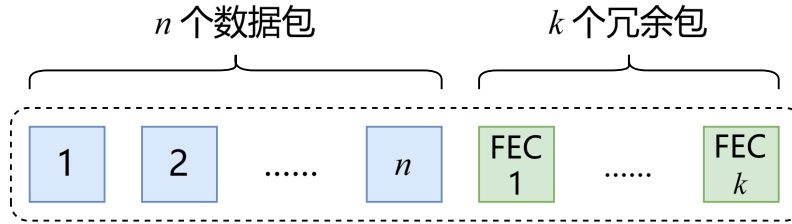


图 3.3 分组码为 n 个数据包附加 k 个冗余包

在 XOR 编码中， n 可以为任意值，而固定 $k = 1$ ，冗余包通过将所有组内的数据包按位进行异或运算得到。如果接收端只接收到了一组数据包共 $n + 1$ 个包中的 n 个，则丢失的包可以通过对已经接收到的包按位进行异或运算恢复得到。XOR 编码可以在一组数据共 $n + 1$ 个包丢失任意一个时通过剩余的 n 个包将丢失的包恢复，但是如果丢失了两个或更多包，则完全不能恢复丢失的数据。XOR 码的计算简单，冗余包生成和丢失数据包恢复都只需要使用异或运算即可完成，运算开销小，但是只能恢复固定模式的少量丢包，面对组内多个丢包的情况效果有限。

为应对 XOR 编码的缺点，在 1960 年，Reed 与 Solomon 提出了 R-S 编码^[12]。R-S 编码保证，对于 n 个数据包和 k 个在有限域上计算出的冗余包共 $n + k$ 个数据包，接收端只要接收到了其中的任意 n 个，就能完整地恢复出所有的原始数据包。相较于 XOR 编码，R-S 编码的恢复能力有较大的提升，能够在同一个编码组里出现较多的丢包的恶劣情况下进行恢复，从能承受最多 1 个丢包增加至能承受最多 k 个丢包。RS 编码被广泛用于传输音视频流媒体，Lin 等人^[26]通过在无线局域网链路上对数据包进行 FEC 编码，提升了视频传输的效果。更多的其他研究者选择结合视频编码自身以帧和画面组（Group of Pictures, GOP）进行编码的特性，进行 FEC 编码以提升视频传输质量。Shih 等人^[27]通过对视频中的关键帧进行 FEC 保护，提升了关键帧以及后续多个依赖关键帧的画面质量，有效提升了视频传输质量。Xiao 等人^[28]使用贪心算法动态决定每个冗余组需要包含的帧数量及冗余度，在不牺牲延迟的情况下提升了视频质量。Yang 等人^[29]通过估算不同的数据包丢包后对解码视频的影响时间，动态选择 FEC 参数以提升用户的视频观看体验。Kurdoglu 等人^[30]则将 FEC 冗余率与编码帧率、编码量化参数及编码方式等联

合优化，以最佳化用户观看体验而非追求更高的单一量化指标。总体而言，XOR 码和 R-S 编码相比简单复制冗余具有更高的冗余恢复效率，因此被广泛应用于实时音视频传输等场景。

然而，R-S 编码通常以数据组为单位进行编码与恢复，其恢复能力依赖于单个编码组中的丢包数量不超过冗余包数量 k 。实际网络上的丢包并不是独立的，在部分链路上可能由于链路拥塞、无线信号衰减等原因出现连续的突发丢包。在这些场景下如果使用 R-S 编码进行丢包恢复，为了能成功恢复数据，必须按照最差的可能情况决定 n 与 k 的相对取值，而这通常使得算法对网络的状态产生过于悲观的估计，为了应对短暂出现的连续丢包而将 k 的值始终维持在较高水平。这导致在其他未遭遇连续丢包的数据组中，大量的冗余包被浪费，占用了传输带宽而未能有效地提升传输质量。为解决此问题，研究者们提出了交织（Interleave）技术。如图 3.4 所示，交织技术将多个编码组交替地在网络上发出，使得当传输过程中出现了连续丢包时，丢包被分散在多个不同的编码组中分别应对，使得单个编码组需要应对的丢包比例大大下降，从而降低了整体需要的冗余率。

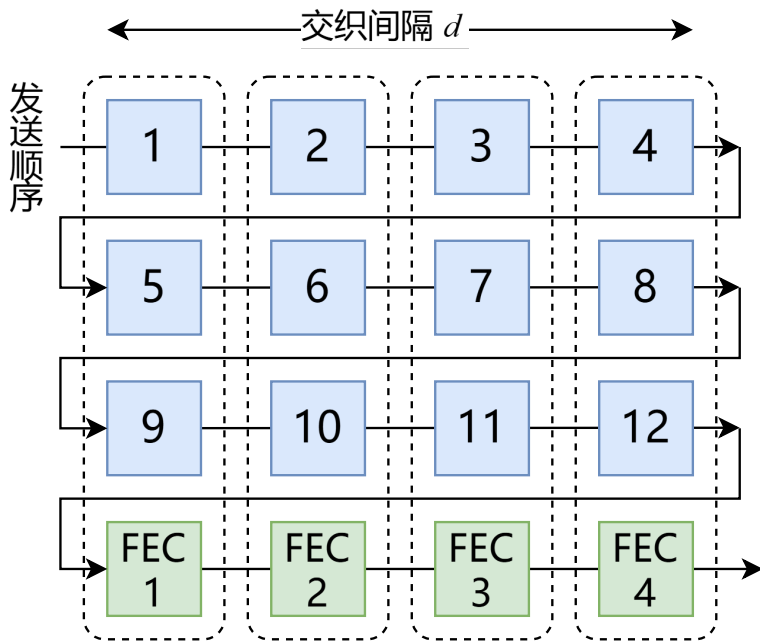


图 3.4 交织编码示意

Liu 等人^[31]提出了一种在开放光通信场景下利用交织应对连续丢包的方法。该方法通过马尔科夫链对网络的状态进行建模，通过测量信道的“开启时间”和“中断时间”，估计信道的连续丢包长度和数据包接收时间特性，同时综合考虑缓冲区分大小、FEC 恢复丢包数量上限等因素，联合优化交织参数和 FEC 参数。Yin 等人^[32]将 FEC 交织编码应用于多跳无线网络的物联网场景中，在每一跳的转发设备

上都利用上游设备的 FEC 编码对发送内容进行恢复后再重新编码发送至下游。作者提出了一种利用力学中势能概念衡量交织性能的方法，同时提出了一种基于此指标对交织参数进行优化的算法。

分组冗余码通过设计比简单复制更复杂的冗余信息计算和丢失包解算机制，允许通过调整参数动态变化冗余率以适应不同丢包率的网络环境。结合交织技术，可以有效地应对真实网络中存在的连续丢包等特性，得到了广泛的应用。

3.2.3 流式冗余码（Streaming 码）

XOR、R-S 等分组码结合交织已经能较好地应对网络中的丢包问题，但是这些编码仍旧不能满足一些实时性需求高的应用。如图 3.5，由于分组码的冗余包通常是通过对所有的组内的数据包进行计算得到，因此冗余信息必须在所有数据包已经发出后才能够计算并在网络中发出，这导致如果接收端在接收数据包时如果检测到了丢包且需要利用冗余信息进行恢复，为了保证数据包按发送顺序连续交付至上层应用，接收端通常需要暂停后续数据的解码输出，直至对应的冗余包到达并完成恢复。由此产生的恢复等待时间会显著增加端到端时延。对于实时性要求较高的应用，即使最终能够恢复出丢失数据，其对应的视频帧或音频数据也可能已经错过播放时限，从而无法有效改善用户体验。

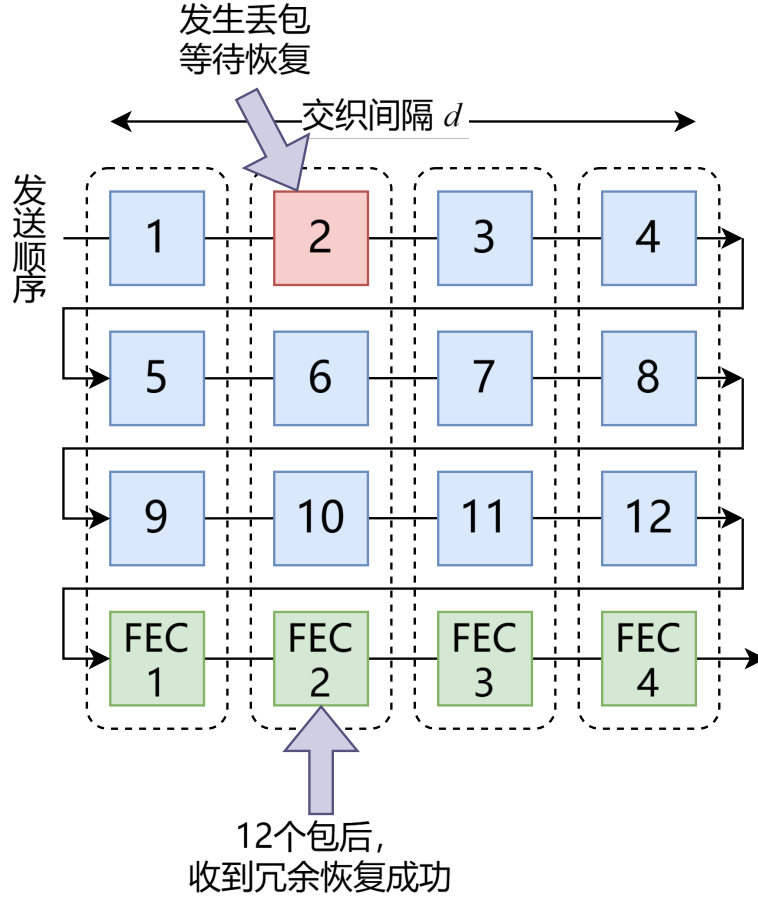


图 3.5 分组码需要暂停解码输出等待冗余包到来才能恢复丢包并继续解码过程

基于此，Martinian 等人提出了流式编码（Streaming Code）^[13]。与传统基于固定编码块的 FEC 不同，流式编码采用跨时间窗口的卷积式编码结构，将同一时刻数据包的红余信息分散嵌入到后续多个时刻发送的数据包中，从而在时间维度上持续提供保护。例如，某一时刻 $t = 0$ 发送的数据，其相关冗余不仅存在于当前数据包中，还会被逐步附加到 $t = 1, 2, 3$ 等后续时刻发送的数据包内。当 $t = 0$ 时刻的数据包发生丢失时，接收端可以利用后续若干时刻收到的数据包逐步恢复其内容，并在预设的有限解码时延内完成恢复，而无需等待整个编码块全部发送完成。该机制能够在保证连续突发丢包恢复能力的同时，显著降低恢复延迟，更适用于实时流媒体等低时延传输场景。Martinian 等人进一步证明了，在给定码率与突发丢包长度条件下，流式编码能够达到理论上的最小恢复时延下界。已经有一些研究工作^[33,34]尝试将流式编码应用于实时音视频通信领域，获得了一定的效果提升。

前向纠错技术逐渐从早期基于简单重复发送的冗余机制，发展到结合有限域运算的分组纠错码，并进一步演化出结合交织技术与时间维度编码的低时延流式编码结构。不同类型的 FEC 机制在冗余开销、连续丢包恢复能力以及恢复时延等方面各有侧重：简单复制具有实现简单、恢复迅速的特点，但冗余效率较低；XOR

码与 R-S 码等分组码能够显著提高冗余恢复效率，但通常需要等待整个编码组完成后才能进行恢复；而流式编码则通过跨时间窗口的连续冗余保护，在保证突发丢包恢复能力的同时进一步降低了解码等待时延，更适用于实时音视频通信等低时延场景。因此，如何在冗余率、恢复能力与恢复时延之间取得平衡，已经成为当前链路质量优化与实时媒体传输中的重要研究方向。

尽管现有 FEC 技术已经能够有效提升低质量网络环境中的数据恢复能力，但大多数研究主要关注编码结构本身的恢复性能、冗余效率以及恢复时延等问题，通常默认数据传输路径已经固定，而较少进一步考虑不同网络链路之间的质量差异与成本差异。在跨域云网络场景下，不同链路可能同时具有显著不同的传输性能与租赁成本，如何结合链路状态动态选择冗余保护策略，并进一步联合流量调度共同优化整体传输性能与网络成本，仍然是值得进一步研究的问题。

3.3 软件定义网络与网络调度

软件定义网络（Software defined networking, SDN）指的是将网络中各个转发设备的数据平面与控制平面解耦，集中进行控制的网络。SDN 网络大大简化了网络的管理和控制流程。对于跨域云网络及其中部署的虚拟网络，尽管有部分的网络设备由 SDN 统一控制，但是各个设备间的跨域互联通常仍仍旧由传统的网络设备提供连接，形成了混合形软件定义网络（hybrid SDN network），如图 3.6^[35]。

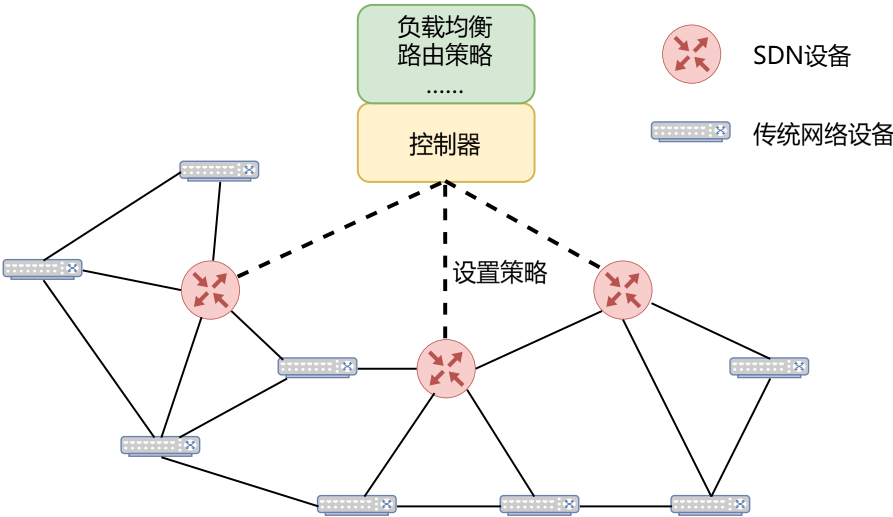


图 3.6 混合 SDN 网络

随着云计算与实时互联网应用的发展，现代云网络中的跨域流量规模持续增长，用户对于传输质量与服务稳定性的要求也不断提高。在跨地域云网络场景中，不同节点之间通常并非只存在单一的物理连接路径，而是可能存在多种不同质量、

不同价格的传输链路。例如，许多云服务商同时提供公网链路互联与专线链路互联^[10,11]。其中，专线链路通常具有更稳定的传输性能、更低的丢包率与时延，但部署成本与使用成本较高；而公网链路虽然成本较低，却容易受到网络拥塞、跨域路由波动等因素影响，出现高丢包、时延抖动等问题^[2]。与此同时，链路质量与网络负载往往还会随着时间动态变化，使得不同链路在不同时间段内呈现出不同的性能特征。

在这种场景下，仅依赖传统网络静态地选择固定传输路径，难以同时满足不同业务对吞吐、时延、可靠性以及成本控制等方面的需求。相比之下，基于 SDN 的覆盖网络能够通过集中控制的方式，对网络中的链路状态、节点负载以及业务需求进行统一管理，并动态地对流量进行调度与路径选择，从而更灵活地利用不同网络资源，在传输性能、可靠性与成本之间取得平衡。因此，如何基于覆盖网络与 SDN 架构实现高效的网络调度，逐渐成为跨域云网络与实时媒体传输领域的重要研究方向。

最初的一些工作主要集中在覆盖网络的建立与路由绕行方面。覆盖网络的概念最初由 Anderson 等人提出^[36]，该工作中介绍了 RON 这一实验性覆盖网络。该工作提出了将公网中并不直接相连的一些节点重新抽象为一个覆盖网络中的相邻节点，称为 RON 节点。各个 RON 节点之间通过公网建立连接，形成 Overlay 网络中的虚拟链路。除了转发功能，RON 节点间还可以通过主动探测的方式，对建立虚拟链路所依靠的物理链路质量进行实时测量，并将测量结果汇总至控制器。当客户端希望通过 RON 网络进行连接时，控制器将综合考虑覆盖网络中所有可用的连接的质量，选择最符合客户端的传输需求的链路对流量进行调度。Roy 等人^[37]根据路由可靠性和 TCP 性能建立指标，并以此为标准优化转发节点的选择。作者提供了多种不同的算法，包括贪心算法、随机算法及两者的混合算法，分别对两种指标存在不同的侧重，供用户灵活根据需要选择。

之后的一些研究进一步研究了通过流量调度实现对资源利用的优化。CRONets^[38]提出了利用云网络服务商提供的虚拟机网络链路建立覆盖网络的方案，并利用多路径 TCP 在覆盖网络节点间提升性能。在覆盖网络资源规模进一步扩大的背景下，研究者开始关注如何通过集中式调度提升资源利用效率。B4^[39]则提出了通过流量调度和流量工程，有效分配不同链路的负载以最大化链路使用率的方法。BDS^[40]使用统一的中央控制器持续监控不同覆盖网络间节点的可用资源，动态调度传输路径以实现链路的带宽的充分利用。除了单纯追求更高链路利用率，一些研究开始进一步联合考虑性能与部署成本之间的平衡。Skyplane^[41]则观察到云网络提供商不同地域资源的定价差异，将追求文件传输最大吞吐量与追求

更低租赁成本建模为一个线性优化问题，给定其中一个指标的限制，利用算法最优化另一个指标。Titan^[2]则针对持续运行的流媒体服务将租用云网络互联资源成本纳入考量，在维持用户体验在一定水平之上的前提下，动态调度流量与计算资源，降低整体的网络部署成本。与此同时，随着实时媒体等对服务连续性要求更高的应用出现，部分工作开始关注链路状态变化时的快速恢复能力。Troia 等人^[42]利用 eBPF 技术实时检测各个覆盖网络节点的传输状态，并在检测到链路拥塞或其他链路质量变化事件时，快速重新触发流量调度算法以维持高质量连接。XRON^[3]则同时结合链路成本优化、资源利用与快速恢复，主动探测可用的公网链路和专线链路质量，结合未来用户流量需求预测，持续计算和更新成本最佳的流量调度策略。为保持所承载音视频通话的服务质量，计算多个备用调度方案以确保故障条件的快速恢复。

3.4 本章小结

本章围绕本文研究问题介绍了三类相关工作。首先，覆盖网络隧道技术通过在底层 IP 网络之上封装二层或三层报文，为云网络中的虚拟互联提供了基础能力。VXLAN、NVGRE 和 Geneve 等协议在封装方式、可扩展性和部署复杂度上各有特点，为构建可控的覆盖网络转发路径提供了工程基础。

其次，链路质量优化相关工作主要利用 FEC 等机制缓解低质量链路上的丢包问题。从简单复制冗余、XOR 码和 R-S 码等分组码，到交织编码和流式编码，已有工作在冗余效率、突发丢包恢复能力和恢复延迟之间进行了不同权衡。这些方法能够提升低质量网络中的丢包恢复能力，但通常以端到端路径为对象，较少考虑跨域云网络中不同链路片段之间的质量差异。

最后，软件定义网络与网络调度相关工作利用集中控制、路径选择和流量工程等方法优化覆盖网络资源使用，并在一定程度上兼顾传输性能和链路成本。然而，这类方法通常倾向于在公网质量下降时规避低质量链路，转而使用专线或其他高质量路径，而较少考虑对低质量公网链路进行修复。与上述工作不同，本文将覆盖网络的分段可控能力与 FEC 链路修复能力结合起来，在全公网互联的条件下对低质量链路片段进行针对性优化。

第 4 章 跨域云网络传输性能提升研究

4.1 系统总体架构

为了实现在全公网互联结构下对低质量链路片段进行针对性修复，本工作主要需要解决三个挑战：

挑战一：如何在用户包大小可变的场景下设计链路片段级 FEC 编码方案。系统作为通用转发平台，承载的上层应用可能产生任意大小的数据包。当用户数据包接近 MTU 时，若 FEC 编码产生的冗余信息追加在用户数据包内一同发送，则封装后的数据包可能超出 MTU，导致 IP 层分片或传输失败。因此，FEC 编码方案必须能够在不影响用户数据包大小的情况下独立添加冗余信息。同时，由于本文只在低质量公网片段上进行修复，编码方案还需要在单个链路片段上有效应对连续突发丢包，而不是依赖端到端重传。本文在 4.2 节设计了交织 XOR 分组编码方案应对此挑战。

挑战二：如何判断差链路所需的冗余强度并自适应地选择编码参数。全公网方案不能简单地对所有链路长期使用高冗余率，否则低成本公网节省下来的费用会被额外流量开销抵消。系统承载的应用中又包含实时音视频流媒体等对延迟敏感的业务，FEC 编码引入的冗余包不仅占用额外带宽，也会引入解码等待延迟。因此，系统需要根据链路片段上的实时丢包统计，在丢包恢复能力、额外带宽开销和恢复延迟之间取得平衡。由于公网链路的丢包率与丢包模式随时间动态变化，固定的编码参数无法同时适应不同质量状态的链路。本文在 4.3 节通过建立丢包信道模型并据此进行约束搜索来解决此挑战。

挑战三：如何消除 FEC 解码按组突发输出对拥塞控制算法的干扰。FEC 解码器按编码组为单位批量恢复和交付数据包。当一个编码组恢复完成后，组内的所有数据包被一次性连续交付给上层应用。这种突发式的输出模式会使得上游的拥塞控制算法收到密集的 ACK 确认包，从而错误地估计链路可用带宽，引发发送速率的震荡。速率震荡不仅影响应用的传输性能，还会使 FEC 编码器的输入节奏不稳定，进一步干扰吞吐量统计和参数估计的准确性。本文在 4.4 节中，在解码端设计了基于 PI 控制器的输出速率控制器来消除此问题。

本系统的整体架构如图 4.1 所示。系统由一个中心控制器（Coordinator）和多个部署在不同地域的转发节点（Node）组成。中心控制器维护节点和连接状态，为端到端连接分配流标识，向相关节点下发转发表项，并根据解码端上报的丢包统计调整低质量链路片段上的 FEC 编码参数。转发节点负责数据面的实际转发，在

本地根据控制器下发的配置，对不同流分别执行普通转发、FEC 编码或 FEC 解码。

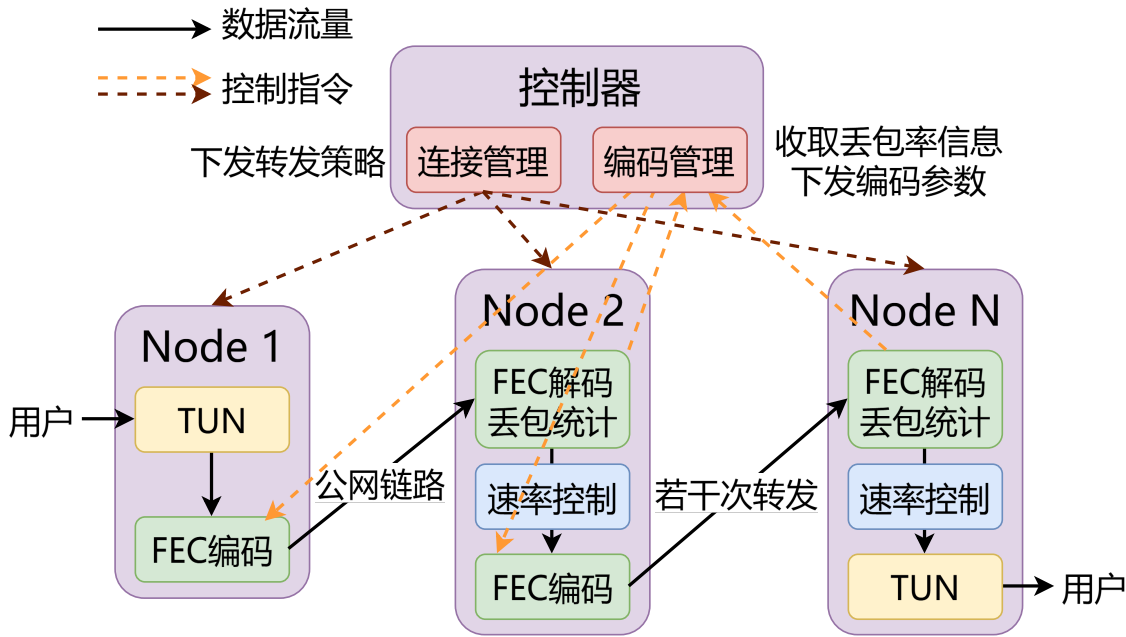


图 4.1 系统总体架构

每个转发节点上为每条端到端连接创建独立的 TUN 虚拟网络设备，用户应用只感知到一条普通 IP 链路，而不需要感知底层覆盖网络和 FEC 机制。节点之间通过 UDP 隧道传输数据包，每个数据包的头部携带 `flow_id` 以标识所属的数据流。系统为每个 `flow_id` 创建独立的处理线程，线程中包含入方向解码器和出方向编码器：当流量进入一个需要修复的低质量链路片段时，出方向编码器添加 FEC 冗余；当流量离开该链路片段时，入方向解码器根据收到的数据包和冗余包恢复丢包。对于不需要修复的链路片段，编码器和解码器退化为普通转发逻辑。

4.2 交织 XOR 前向纠错编码设计

由于覆盖网络中承载的流量种类多样，

如第二章所述，现有的前向纠错编码方案主要包括简单复制冗余、XOR 码、R-S 码以及流式编码等。本文选择基于 XOR 运算的分组编码结合交织技术作为 FEC 编码方案，其核心考量如下。

分组码天然适应可变包大小的场景。在分组码中，冗余包是独立于数据包的单独包：编码器先将用户数据包逐个作为数据包发出，在编码组填满或超时而再生成独立的冗余包并追加发送。由于冗余信息不嵌入在用户数据包内部，用户数据包的大小不受 FEC 编码的影响，因此即使数据包本身已接近 MTU，也不会因为 FEC 而产生封装溢出的问题。相比之下，流式编码将同一时刻的冗余信息分散嵌

入后续多个数据包中，要求在数据包内部预留冗余空间，在用户包大小不可控的通用转发场景下难以适用。分组码的另一个优势是边界清晰：编码器和解码器可以部署在某一段公网链路的两端，只修复该链路片段，而不会要求整条端到端路径都采用同一种传输机制。

在多种分组码中，本文选择 XOR 编码而非 R-S 码，理由是结合交织技术的 XOR 编码已足以应对公网链路上观察到的丢包模式。根据第一章的分析，公网链路的主要丢包特征是偶发的孤立丢包和有限长度的连续突发丢包。交织技术将连续的突发丢包分散到不同的恢复列中，使得每列至多丢失一个数据包，恰好匹配 XOR 编码“每列可恢复一个丢包”的能力。同时，XOR 编码的编码和解码均只需要按位异或运算，无需有限域上的矩阵运算，计算开销极低，适合高吞吐量的转发场景。

具体地，本文提出的交织 FEC 编码将数据包组织为一个二维矩阵结构，如图 4.2 所示。设交织深度为 d ，保护包数为 k ，则每个编码组包含 $d \times k$ 个数据包和 d 个冗余包。矩阵共有 d 列、 $k+1$ 行，其中前 k 行为数据包，第 $k+1$ 行为冗余包。每个数据包在矩阵中的位置由其组内序列号唯一确定：对于序列号为 s 的数据包，其所在列号为 $s \bmod d$ ，行号为 $\lfloor s/d \rfloor$ 。每个冗余包通过对同一列中所有数据包进行按位异或运算得到。交织技术的关键优势在于：当网络上发生长度不超过 d 的连续丢包时，由于相邻数据包被分配到不同的列中，这些丢失的数据包被分散到最多 d 个不同的列中，每个列至多丢失一个数据包，因此每个列都可以独立恢复。以图 4.2 为例，当 $d = 4$ 时，即使出现了连续四个丢包，由于四个丢包位于四个不同的编码组中（图中以虚线框标出），在每个编码组内部只有一个包丢失，因此接收端仍旧可以完全恢复所有的丢包内容。

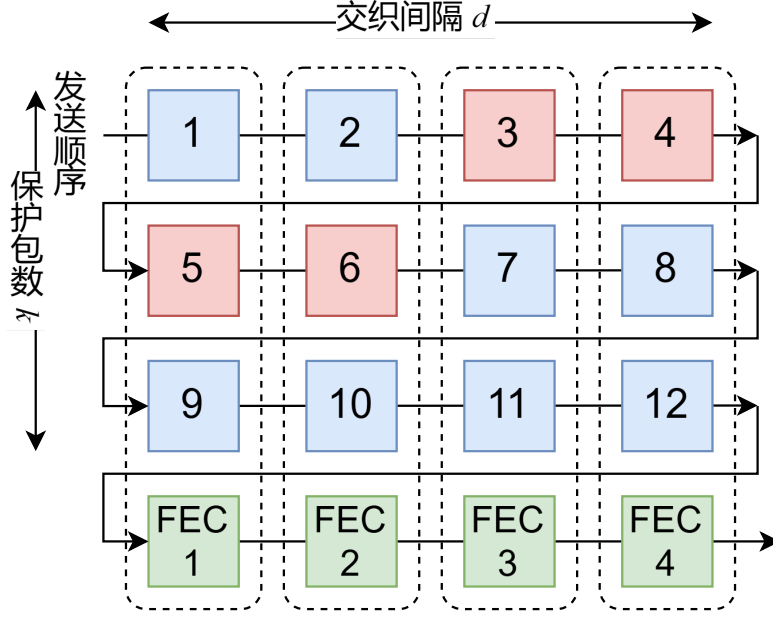


图 4.2 交织编码矩阵结构示意图 ($d = 4, k = 3$)

编码参数 (d 和 k) 可以在编码组之间动态切换, 编码器在结束当前组时加载最新收到的参数配置, 下一编码组立即使用新参数。

4.3 基于丢包统计的自适应参数调整

由于不同公网链路的质量差距大, 且单个链路的链路质量也会随着时间变化, 本文希望仅在差链路上使用足够但不过量的冗余, 以达成对公网链路的充分利用。因此, FEC 编码参数需要同时满足多种应用的需求, 既包括流媒体应用的延迟需求、也包括文件传输等应用的带宽需求、控制额外带宽开销, 并适应链路质量的动态变化。本文的解决思路是: 首先为公网链路的丢包行为建立一个数学模型, 然后从接收端的丢包观测量中估计模型参数, 最后在延迟与残余丢包率的约束下搜索最优编码参数。

4.3.1 丢包信道模型

根据第一章中对公网链路丢包特性的分析, 公网链路上的丢包行为可以大致分为两类: 一类是偶发的孤立丢包, 丢失一个包后链路随即恢复正常; 另一类是连续的突发丢包, 由于链路拥塞等原因连续丢失多个数据包。基于这一观察, 本文提出一个简化的三状态丢包信道模型, 如图 4.3 所示。模型定义三个状态: S_1 (孤立丢包状态, 丢失一个包后立即恢复)、 S_2 (正常状态, 当前数据包正常接收)、 S_3 (连续丢包状态, 连续丢失多个包)。模型通过三个参数 p_{21} (孤立丢包触发概率)、 p_{23} (连续丢包触发概率) 和 p_{33} (突发延续概率) 完整描述链路的丢包行为。

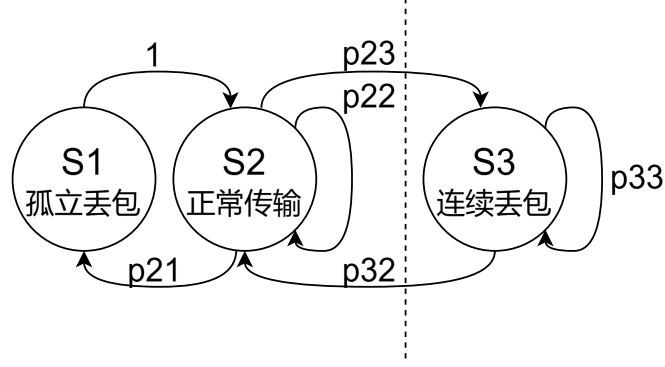


图 4.3 三状态丢包信道模型

4.3.2 参数估计与编码参数搜索

解码端通过全局序列号间隔检测丢包，维护一个滑动窗口统计近期的丢包事件，并将统计量定期上报至中心控制器。中心控制器收到统计量后，首先根据丢包事件的突发长度分布估计三状态模型的参数 p_{21} 、 p_{23} 和 p_{33} ，进而得到丢包事件的总发生率 $\lambda = p_{21} + p_{23}$ 。

在估计出模型参数后，中心控制器在给定的性能约束下搜索最优的编码参数 (d, k) 。算法考虑交织深度 $d \in \{1, 2, 3, 4\}$ 的候选值，对于每个 d ，确定满足以下两个约束的最大保护包数 k ：

1. **延迟约束：**编码组引入的额外等待延迟不应超过阈值，以满足流媒体应用的实时性需求。
2. **残余丢包率约束：**编码后未被恢复的随机丢包率应低于阈值，确保应用层的丢包率在可接受范围内。

最终在所有可行的 (d, k) 组合中，选择 k 值最大的组合作为最优编码参数，在提供充足保护的前提下尽可能减少冗余带来的额外开销。上述参数调整过程构成了一个完整的反馈闭环：解码端持续检测丢包并上报统计信息，中心控制器计算最优参数并下发至编码端，编码端在下一编码组生效新参数。

4.4 解码端输出速率控制设计

本节针对挑战三，介绍解码端的输出速率控制器（Pacer）设计。如前所述，FEC 解码器按编码组为单位批量交付数据包，这种突发输出会使上游 CCA 收到密集的 ACK，导致错误的带宽估计和速率震荡。本文通过在解码端引入 PI 速率控制器，将突发输出平滑为匀速流来解决此问题，其控制模型如图 4.4 所示。

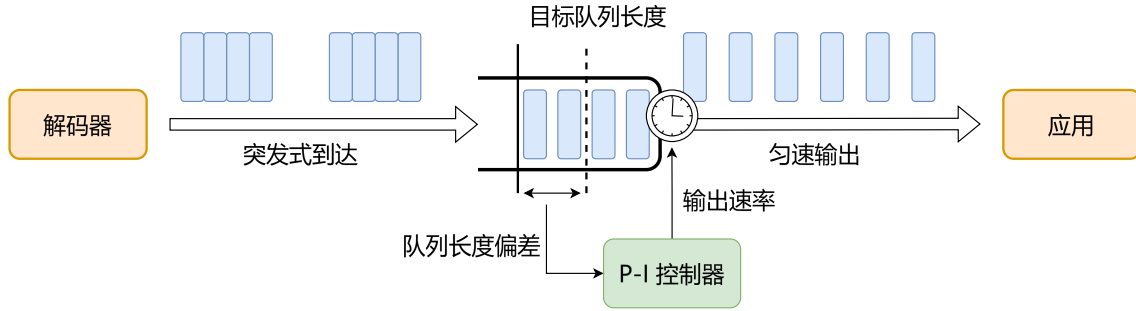


图 4.4 Pacer 控制模型

输出速率控制器的核心是一个基于缓冲区深度的 PI（比例-积分）控制器。FEC 解码器恢复后的数据包首先进入一个输出缓冲区，输出速率控制器根据缓冲区的当前深度与目标深度的偏差计算出发包速率，并按照该速率匀速地从缓冲区中取出数据包交付给上层应用。比例项提供快速的瞬态响应：缓冲区超过目标时加快发包消耗积压，低于目标时减慢发包等待新数据。积分项消除稳态误差：即使数据包到达速率发生变化，积分项也能逐步学习新的稳态速率，确保缓冲区深度收敛到目标值。启动时，输出速率控制器先以直通模式运行并测量实际的到达速率，待积累足够的观测后切换到 PI 控制模式，切换时将测量速率设置为积分项的初始值以实现平滑过渡。

具体来说，控制器内部维护一个目标缓冲区深度 h_0 ，它是 5 个数据包的深度或是按当前释放速率计算 5 ms 能排空的缓冲区大小，取二者较大者。同时，控制器内部维护了一个积分值 I ，它在从直通模式切换至 PI 控制模式时初始化为在直通模式下估计的包到来速率。每次有数据包从缓冲区中释放时，控制器计算当前的缓冲区深度 h 与目标深度的差值：

$$\Delta h = h - h_0 \quad (4.1)$$

同时记录自从上次释放数据包至当前时刻经历的时间 Δt ，并由上一时刻的积分值 I_{prev} 计算新的积分值：

$$I = I_{\text{prev}} + \Delta h \Delta t \quad (4.2)$$

进一步决定当前的输出速率 v （以包/秒为单位）：

$$v = K_p \Delta h + K_i I \quad (4.3)$$

其中 K_p 与 K_i 为可以调整的参数。

在稳态时，缓冲区中的包到来速度与释放速度相等，缓冲区的长度始终稳定在 h_0 附近， Δh 接近 0，包的输出速度由稳定的 I 决定；当包的到来速度增长时，包从缓冲的释放速度低于包的到来速度增长，缓冲的长度开始增长，长度超过 h ，

使得 Δh 为正, 进而使得 I 增长。这共同使得 v 提升以跟随包到来速度的增长。当包到来速度下降时, 类似的机制促使 v 也跟随下降, 直到输出速率再次达到平衡。通过此设计, 包的从缓冲区的离开速率总能匹配包到来的平均速率。

输出速率控制器与 FEC 编码的自适应参数调整形成了协同关系。输出速率控制器将突发输出平滑为匀速流, 使得拥塞控制算法能够基于稳定的 ACK 反馈正确估计带宽, 维持稳定的发送速率。稳定的发送速率使得 FEC 编码器以稳定的节奏填满编码组, 进而使吞吐量统计值更加准确, 提升了参数调整的准确性。

4.5 本章小结

本章围绕全公网互联、差分片修复的总体思路, 介绍了本文提出的跨域公网链路传输优化方法。首先明确了本文不依赖专线互联, 而是在公网覆盖网络中针对低质量链路片段进行 FEC 修复; 随后介绍了集中控制、分布转发的系统总体架构, 并将总体目标细化为三个设计挑战。针对通用转发场景下可变包大小与 MTU 约束的挑战, 本文选择了交织 XOR 分组编码方案, 冗余包作为独立包发送不影响用户数据包大小, 交织技术则将连续丢包分散到不同的恢复列中。针对满足实时应用延迟需求的同时自适应选择编码参数的挑战, 本文提出了三状态丢包信道模型, 介绍了从丢包观测量估计模型参数以及在延迟与残余丢包率约束下搜索最优编码参数的方法。最后, 针对 FEC 解码突发输出干扰拥塞控制算法的挑战, 本文设计了解码端 PI 速率控制器, 将突发输出平滑为匀速流, 并与 FEC 参数调整机制形成协同。

第 5 章 实验验证与分析

本章主要介绍系统的实现情况及实验结果。?? 节介绍本文提出的设计方案的实现情况及实验环境与实验设置等基本信息，5.2 节展示了实验结果，并对其进行了简要的分析。

5.1 实验环境

本文作者使用 Rust 语言实现了设计的分布式转发与控制系统，实现了动态 FEC 编解码、丢包统计及参数自动计算等所有核心设计要点。本文实验在一台配备两颗 Intel Xeon E5-2620 v3 处理器的服务器上进行，整机共提供 12 个物理核心、24 个逻辑 CPU。服务器共搭载 64 GiB 运行内存。本文中的实验使用 Rattan^[43] 对网络上的丢包，延迟及带宽限速进行模拟。在实验中，多台不同的虚拟机通过虚拟交换机互联，并在每台虚拟机的虚拟出口使用 Rattan 对网络流量进行处理，以达成对链路延迟、丢包带宽等不同性质的模拟。如图 5.1，在实验中，共使用三台虚拟机（记作 A，B，C）作为转发以及用户接入客户端，使用另外一台虚拟机作为控制器。利用控制器建立 A 经由 B 到达 C 的链路，其中 AB 间的链路为模拟低质量链路，存在丢包，往返时延 50 ms；BC 间的链路为模拟高质量链路，为不丢包的链路，往返时延 50 ms。两条链路的带宽均为 100 Mbps。通过调整 AB 间链路的丢包率并测量 AC 之间的传输性能，可以测量本文提出的方法与直接进行转发的基准方案的性能，并进行对比。

实验时，分别选取丢包率取从 0 至 2% 的不同值，分别使用本文提出的动态链路优化算法与直接进行转发的方法在 A、C 间使用 iperf3 进行测速，持续 60 s，并记录收端的接收速率。

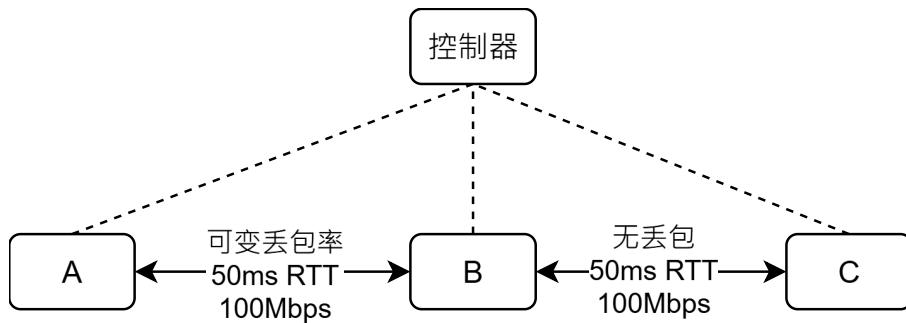


图 5.1 实验建立的网络拓扑

5.2 实验结果与分析

实验结果如图 5.2 所示。图 5.2(a) 展示了不同丢包率下两种方法的端到端吞吐量，图 5.2(b) 展示了本文方法相对于直接转发基准方案的吞吐提升程度。

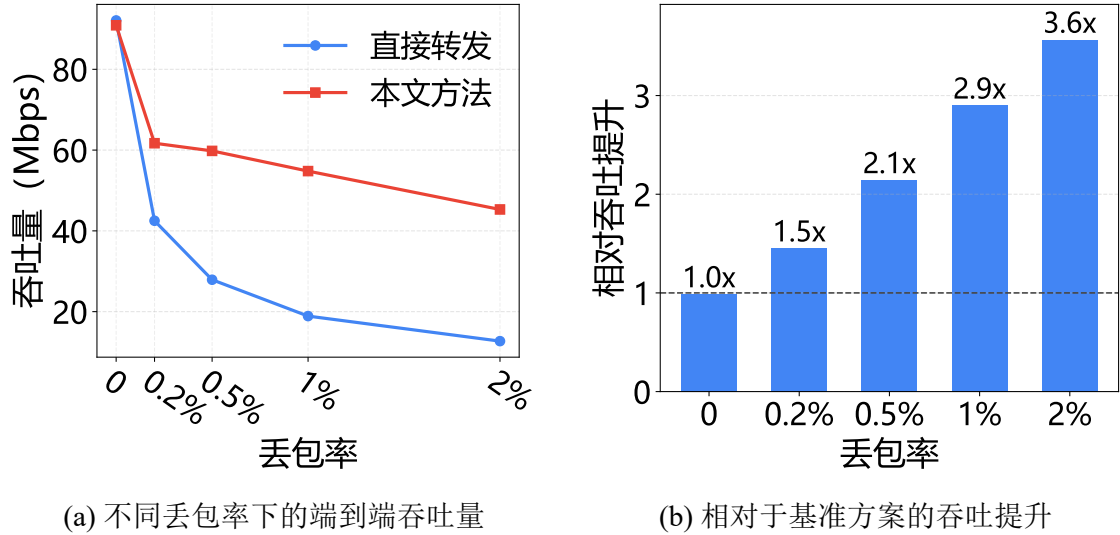


图 5.2 不同丢包率下本文方法的吞吐性能提升

与基线方法相比，本文方法在较高丢包率下仍旧保持了较高的总体带宽，并达到了最高 3.6 倍带宽提升。通过查阅控制器日志可以发现，本方法正确识别了两条链路中的不同丢包率，并在丢包严重的 AB 段链路上启用了 FEC，在没有丢包的链路上则始终使用普通转发。

尽管本方法使用了 FEC 方法对链路丢包进行了恢复，并有效提升了整体吞吐量，但是相比无丢包时仍旧有较大的吞吐性能差距。这一方面是由于添加冗余信息占用了一部分带宽，导致理想状态下应用可用的带宽下降；另一方面是由于冗余参数的选取较为保守，即使添加了 FEC 也有一定概率会出现丢包，导致发送端仍旧不能保证没有丢包发生，这些丢包会导致发送端丢包降低发送速率。最终，冗余带宽占用和残余丢包两种因素叠加导致即使使用了本方法，传输速率仍旧有所下降。

值得注意的是，虽然从本方法的吞吐量从无丢包至有丢包有较大下降，但是随着丢包率的增高，吞吐量的下降程度明显小于直接转发行为，这是由于本方法的动态 FEC 参数决定算法动态根据测量得到的丢包率参数决定了合适的冗余度，从而有效地将修复后的丢包率维持在较为固定的范围内，因而尽管底层链路的丢包率持续恶化，但是上层 TCP 连接感知的丢包率却始终稳定，使得最终的吞吐量也稳定在较为稳定的范围内。

在丢包率为零的场景下，使用直接转发能达到比使用本方法略高的性能，这

主要是由于本方法所使用的 FEC 算法在不启用冗余时也仍旧需要在数据包中传输少量额外的包头，导致吞吐量相比直接转发有大约 1 % 的下降，与存在丢包场景下的性能提升相比影响较小。

第 6 章 结论与展望

6.1 工作总结

随着云计算和实时互联网应用的发展，跨地域覆盖网络需要在全球范围内为用户提供稳定的低延迟、高带宽和低丢包传输服务。传统方案通常依赖专线链路保证服务质量，但专线链路价格较高，难以满足大规模跨域覆盖网络的成本优化需求。已有链路调度类方法尝试在公网质量较好时将部分流量迁移至公网链路，以降低专线链路上的承载流量；已有冗余编码类方法则通过端到端前向纠错编码缓解网络丢包对应用体验的影响。然而，本文的测量和分析表明，公网链路质量下降时段往往与用户流量高峰重合，使得基于公网分流的调度方法难以有效削减专线峰值成本；同时，跨域覆盖网络中不同公网链路片段质量差异显著，端到端统一添加冗余会在质量良好的片段上引入不必要的带宽开销。

针对上述问题，本文提出了一种面向跨域公网链路的分段质量修复方法。本文方法不再依赖专线链路作为后备方案，而是在全公网互联的覆盖网络中识别和修复低质量链路片段。对于质量良好的链路片段，系统保持普通转发，以避免额外开销；对于丢包率较高、存在连续突发丢包的链路片段，系统在片段两端加入前向纠错编码与丢包恢复机制，从而将质量修复限制在真正发生问题的网络范围内。该思路充分利用了覆盖网络中间节点可控、路径可分段的特点，在降低链路成本的同时提升公网链路的有效传输质量。

围绕这一思路，本文设计并实现了一套分布式覆盖网络转发与链路优化系统。系统采用中心控制器和分布式转发节点相结合的架构，由控制器维护连接状态、下发转发表项并计算 FEC 参数，由转发节点执行普通转发、FEC 编码和 FEC 解码。为适应通用覆盖网络中用户包大小可变的特点，本文设计了交织 XOR 分组编码方案，使冗余包以独立数据包形式发送，避免修改用户报文内容；为适应公网链路质量的动态变化，本文建立三状态丢包信道模型，并根据接收端上报的丢包统计动态选择交织深度和保护包数；为避免 FEC 解码按组恢复造成突发输出，本文进一步设计了基于 PI 控制器的输出速率控制机制，使解码后的数据包以更平滑的节奏交付给上层传输协议。

最后，本文使用 Rust 语言实现了上述系统，并在模拟跨域低质量链路的实验环境中对本文方法进行了验证。实验结果表明，在 0 至 2% 的链路丢包率范围内，本文方法能够正确识别低质量链路片段并在该片段启用 FEC 修复，在无丢包链路片段上保持普通转发。与直接转发方案相比，本文方法在存在丢包时显著提升了

端到端吞吐量，最高实现了约 3.6 倍的吞吐提升。实验也表明，本文方法在无丢包场景下仅引入较小的额外开销，而在低质量公网链路场景下能够有效缓解丢包对传输性能的影响，验证了分段链路质量修复思路的有效性。

6.2 未来工作展望

本文提出的分段链路质量修复方法为全公网覆盖网络提供了一种降低专线依赖、提升跨域传输质量的思路，但仍有若干方向值得进一步研究。

首先，可以进一步优化 FEC 参数选择算法。本文当前的参数调整方法基于接收端上报的丢包统计估计链路丢包模型，并在延迟约束和残余丢包率约束下选择编码参数。后续工作可以引入更精细的链路状态建模方法，进一步区分孤立丢包、短突发丢包和长突发丢包等不同模式，并结合在线反馈机制持续修正模型参数。通过更准确地匹配链路的实际丢包特征，系统可以在不显著增加冗余开销的前提下进一步降低 FEC 恢复失败的概率，从而提高修复后链路质量的稳定性。

其次，可以在链路片段层面引入轻量级重传机制。FEC 能够在不等待端到端反馈的情况下恢复大部分丢包，但在丢包模式超出当前编码参数保护能力时，仍可能出现少量残余丢包。由于本文系统将修复逻辑部署在相邻覆盖网络节点之间，链路片段的往返时延通常显著小于完整端到端路径的往返时延。因此，未来可以在 FEC 恢复之后，针对剩余的少量未恢复数据包触发链路级快速重传，以较短的链路局部反馈时延补充 FEC 的恢复能力。FEC 与链路级重传相结合，有望在保持较低恢复延迟的同时进一步降低应用层感知到的残余丢包率。

参考文献

- [1] AppLogic Networks. The 2026 Global Internet Phenomena Report[R/OL]. AppLogic Networks, 2026[2026-05-26]. <https://www.applogicnetworks.com/gipr-2026>.
- [2] Kataria B, Lnu P, Bothra R, et al. Saving private wan: Using internet paths to offload wan traffic in conferencing services[J]. Proceedings of the ACM on Networking, 2024, 2(CoNEXT4): 1-22.
- [3] Wu B, Qian K, Li B, et al. Xron: A hybrid elastic cloud overlay network for video conferencing at planetary scale[C]//Proceedings of the ACM SIGCOMM 2023 Conference. 2023: 696-709.
- [4] Ha S, Rhee I, Xu L. Cubic: a new tcp-friendly high-speed tcp variant[J]. ACM SIGOPS operating systems review, 2008, 42(5): 64-74.
- [5] Cardwell N, Cheng Y, Gunn C S, et al. Bbr: Congestion-based congestion control: Measuring bottleneck bandwidth and round-trip propagation time[J]. Queue, 2016, 14(5): 20-53.
- [6] Arun V, Balakrishnan H. Copa: Practical {Delay-Based} congestion control for the internet [C]//15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18). 2018: 329-342.
- [7] Bolot J C, Fosse-Parisis S, Towsley D. Adaptive fec-based error control for internet telephony[C/OL]//IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320): Vol. 3. 1999: 1453-1460 vol.3. DOI: 10.1109/INFCOM.1999.752166.
- [8] Huang T Y, Huang P, Chen K T, et al. Could skype be more satisfying? a qoe-centric study of the fec mechanism in an internet-scale voip system[J]. IEEE Network, 2010, 24(2): 42-48.
- [9] Holmer S, Shemer M, Paniconi M. Handling packet loss in webrtc[C]//2013 IEEE international conference on image processing. IEEE, 2013: 1860-1864.
- [10] Microsoft. Azure bandwidth pricing[EB/OL]. Microsoft, 2026[2026-05-15]. <https://azure.microsoft.com/en-us/pricing/details/bandwidth/>.
- [11] Google. Google cloud bandwidth pricing[EB/OL]. Google, 2026[2026-5-15]. <https://cloud.google.com/vpc/network-pricing>.
- [12] Reed I S, Solomon G. Polynomial codes over certain finite fields[J]. Journal of the society for industrial and applied mathematics, 1960, 8(2): 300-304.
- [13] Martinian E, Sundberg C E. Burst erasure correction codes with low decoding delay[J]. IEEE Transactions on Information theory, 2004, 50(10): 2494-2502.
- [14] Aliyun. 跨境云企业网价格计算器[EB/OL]. Aliyun, 2026[2026-5-19]. https://www.aliyun.com/price/product#/commodity/cbn_bwp_pre_mkt.
- [15] Tencent. 云网络计费总览[EB/OL]. Tencent, 2026[2026-5-19]. <https://cloud.tencent.com/document/product/877/18676>.

- [16] Azodolmolky S, Wieder P, Yahyapour R. Cloud computing networking: Challenges and opportunities for innovations[J]. IEEE Communications Magazine, 2013, 51(7): 54-62.
- [17] Luong N C, Wang P, Niyato D, et al. Resource management in cloud networking using economic analysis and pricing models: A survey[J]. IEEE Communications Surveys & Tutorials, 2017, 19(2): 954-1001.
- [18] Mahalingam M, Dutt D, Duda K, et al. Request for comments: No. 7348 Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks[M/OL]. RFC Editor, 2014. <https://www.rfc-editor.org/info/rfc7348>. DOI: 10.17487/RFC7348.
- [19] Garg P, Wang Y S. Request for comments: No. 7637 NVGRE: Network Virtualization Using Generic Routing Encapsulation[M/OL]. RFC Editor, 2015. <https://www.rfc-editor.org/info/rfc7637>. DOI: 10.17487/RFC7637.
- [20] Gross J, Ganga I, Sridhar T. Request for comments: No. 8926 Geneve: Generic Network Virtualization Encapsulation[M/OL]. RFC Editor, 2020. <https://www.rfc-editor.org/info/rfc8926>. DOI: 10.17487/RFC8926.
- [21] Microsoft. Network virtualization using generic routing encapsulation (nvgre) task offload [EB/OL]. Microsoft Learn, 2023[2026-05-13]. <https://learn.microsoft.com/en-us/windows-hardware/drivers/network/network-virtualization-using-generic-routing-encapsulation--nvgre--task-offload>.
- [22] OVN Project. General — ovn documentation[EB/OL]. OVN Project, 2026[2026-05-13]. <https://docs.ovn.org/en/latest/faq/general.html>.
- [23] VMware. Nsx-t: Routing where you need it (multi-hypervisor & multi-cloud)[EB/OL]. VMware, 2017[2026-05-13]. <https://blogs.vmware.com/networkvirtualization/2017/09/nsx-t-routing-where-you-need-it.html/>.
- [24] Eddy W. Request for comments: No. 9293 Transmission Control Protocol (TCP)[M/OL]. RFC Editor, 2022. <https://www.rfc-editor.org/info/rfc9293>. DOI: 10.17487/RFC9293.
- [25] Gandikota V R, Tamma B R, Murthy C S R. Adaptive fec-based packet loss resilience scheme for supporting voice communication over ad hoc wireless networks[J/OL]. IEEE Transactions on Mobile Computing, 2008, 7(10): 1184-1199. DOI: 10.1109/TMC.2008.42.
- [26] Lin C H, Shieh C K, Hwang W S. An access point-based fec mechanism for video transmission over wireless lans[J]. IEEE Transactions on Multimedia, 2012, 15(1): 195-206.
- [27] Shih C H, Kuo C I, Chou Y K. Frame-based forward error correction using content-dependent coding for video streaming applications[J]. Computer Networks, 2016, 105: 89-98.
- [28] Xiao J, Tillo T, Lin C, et al. Dynamic sub-gop forward error correction code for real-time video applications[J]. IEEE Transactions on Multimedia, 2012, 14(4): 1298-1308.
- [29] Yang X, Zhu C, Li Z, et al. Unequal loss protection for robust transmission of motion compensated video over the internet[J]. Signal Processing: Image Communication, 2003, 18(3): 157-167.
- [30] Kurdoglu E, Liu Y, Wang Y. Perceptual quality maximization for video calls with packet losses by optimizing fec, frame rate, and quantization[J]. IEEE Transactions on Multimedia, 2017, 20(7): 1876-1887.

- [31] Liu J, Zhang X, Blow K, et al. Performance analysis of packet layer fec codes and interleaving in fso channels[J]. *Iet Communications*, 2017, 11(13): 2042-2048.
- [32] Yin H H, Ng K H, Zhong A Z, et al. Intrablock interleaving for batched network coding with blockwise adaptive recoding[J]. *IEEE Journal on Selected Areas in Information Theory*, 2021, 2(4): 1135-1149.
- [33] Emara S, Fong S L, Li B, et al. Low-latency network-adaptive error control for interactive streaming[J]. *IEEE Transactions on Multimedia*, 2021, 24: 1691-1706.
- [34] Rudow M, Yan F Y, Kumar A, et al. Tambur: Efficient loss recovery for videoconferencing via streaming codes[C]//20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). 2023: 953-971.
- [35] Amin R, Reisslein M, Shah N. Hybrid sdn networks: A survey of existing approaches[J]. *IEEE Communications Surveys & Tutorials*, 2018, 20(4): 3259-3306.
- [36] Andersen D, Balakrishnan H, Kaashoek F, et al. Resilient overlay networks[C]//Proceedings of the eighteenth ACM symposium on Operating systems principles. 2001: 131-145.
- [37] Roy S, Pucha H, Zhang Z, et al. On the placement of infrastructure overlay nodes[J]. *IEEE/ACM Transactions on networking*, 2009, 17(4): 1298-1311.
- [38] Cai C X, Le F, Sun X, et al. Cronets: Cloud-routed overlay networks[C]//2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2016: 67-77.
- [39] Jain S, Kumar A, Mandal S, et al. B4: Experience with a globally-deployed software defined wan[J]. *ACM SIGCOMM Computer Communication Review*, 2013, 43(4): 3-14.
- [40] Zhang Y, Jiang J, Xu K, et al. Bds: A centralized near-optimal overlay network for inter-datacenter data replication[C]//Proceedings of the Thirteenth EuroSys Conference. 2018: 1-14.
- [41] Jain P, Kumar S, Wooders S, et al. Skyplane: Optimizing transfer cost and throughput using {Cloud-Aware} overlays[C]//20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). 2023: 1375-1389.
- [42] Troia S, Mazzara M, Savi M, et al. Resilience of delay-sensitive services with transport-layer monitoring in sd-wan[J]. *IEEE Transactions on Network and Service Management*, 2022, 19(3): 2652-2663.
- [43] Wang M, Shen Y, Wang B, et al. Rattan: An extensible and scalable modular internet path emulator[A/OL]. 2025. arXiv: 2507.08134. <https://arxiv.org/abs/2507.08134>.

致 谢

感谢王博老师、徐明伟老师和黄永峰老师对我论文的悉心指导。他们对我的研究工作给出了很多建议和帮助。

感谢实验室的沈逸昕师兄，他几年来毫无保留地将自己的经验与知识与我分享，让我少走了很多弯路。还需要感谢佟海轩师兄、李骋昊师兄、张皓晨同学和谢雨桐同学，他们在我的研究过程中与我的讨论开拓了我的思路，也让我受益匪浅。

四年的本科时光随着一本毕业论文行至致谢也已行至结尾。四年里我在清华园中遇到了的各位的老师与同学，感谢他们对我的帮助与支持！

声 明

本人郑重声明：所呈交的综合论文训练论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签 名：_____ 日 期：_____