DOI:10.19651/j.cnki.emt.2415489

基于时延程策略的多会话时延损伤模拟*

吴 靖 曹炳尧
(上海大学特种光纤与光接入网重点实验室 上海 200444)

摘 要：随着卫星网络、车联网络、工业网络等业务仿真模拟需求的日益增长,针对传统专用信道损伤仪存在的模拟链路数量少、灵活性低、资源占用高等问题,本文提出一种基于时延量程策略的多会话时延损伤模拟方法,构建灵活的软件网络损伤模拟。该方法通过识别检测独立控制各会话流的时延损伤,并采用基于时延程策略的多队列合并架构以降低资源占用。实验结果表明,相较于传统专用设备与模拟软件 NetEm,该方法支持百万级链路的独立时延配置,会话流数从十数条增加到百万条,且在各带宽下降低至少 85%的内存占用,满足大规模和精度的同时极大的降低系统成本。

关键词：信道时延损伤模拟;多会话流;Jenkins 哈希;时延量程;队列合并
中图分类号：TP391 **文献标识码**：A **国家标准学科分类代码**：510.4

Multi-session delay damage simulation based on time delay strategy

Wu Jing Cao Bingyao
(Key Laboratory of Specialty Fiber and Optics Access Networks, Shanghai University, Shanghai 200444, China)

Abstract: With the increasing demand for satellite network, vehicle-connected network, industrial network and other service simulation, this paper proposes a multi-session delay damage simulation method based on delay range strategy to build flexible software network damage simulation, aiming at the problems of small number of analog links, low flexibility and high resource occupation of traditional dedicated channel damage instruments. In this method, the delay damage of each session flow is identified and controlled independently, and the multi-queue merging architecture based on time delay strategy is adopted to reduce the resource consumption. The experimental results show that compared with the traditional dedicated device and simulation software NetEm, the proposed method supports the independent delay configuration of million-level links, increases the number of session streams from ten to one million, and reduces the memory consumption by at least 85% under each bandwidth, which meets the requirements of large scale and accuracy, and greatly reduces the system cost.

Keywords: channel delay damage simulation;multi-session flow;Jenkins hash;delay range;queue merging

0 引 言

卫星网络是依靠卫星的信号中继能力构成的智能化体系,其将位于不同轨道的卫星以及地面节点通过卫星链路进行互联互通[1]。卫星网络建设作为国家战略中的重要一环,已被视为影响全球发展竞争格局的关键[2]。相比于地面网络,其具有更高的带宽和全球覆盖能力[3],被广泛应用于导航[4]、地球勘测[5]、应急救援[6]、数据采集[7]以及国防军事等领域。同时,随着智能网联汽车的发展[8],智能工业互联网的兴盛,使得针对各项业务的大规模真实信道模拟需求日益增加。时延作为信道模拟中的重要性能指标,能够准确的反应信道链路中的通信质量,因此越来越多针对时延的信道模拟与链路损伤研究成为热点。

目前的时延链路损伤方案主要包括基于硬件实现和软件实现两种类型。基于硬件实现的物理设备采用专用的集成电路(application specific integrated circuit,ASIC)或现场可编程逻辑门阵列(field programmable gate array,FPGA),从而实现高精度地模拟复杂网络链路。例如已经投入使用的思博伦 Attreo 系列网络损伤仪能够实现 100 G 线速下 80 ms 的时延控制,支持最高 8 个独立链路的损伤配置、迈思源的 HoloWAN pro 系列支持可控量程 10 s 精度 4 ns 的时延控制,一对物理网口间最高 15 条独立链路

收稿日期:2024-02-05
* 基金项目:国家重点研发计划(2021YFB2900800)、上海市科委项目(22511100902,22511100502)资助

的时延配置、信而泰 Xcompass-S 系列支持纳秒级精度 10 G 线速 800 ms 的时延控制,最高 8 个独立损伤应用场景的时延配置等。这些专用硬件设备受制于内存与内部资源的限制,导致无法进行大规模会话流的时延链路损伤模拟。同时成本高昂、灵活性差,无法满足通用适应性场景的需求。基于软件实现的模拟工具例如 NetEm[9] 由于依赖于内核协议栈,通常难以达到线速的转发要求,且在处理短帧时性能受限严重[10]。

为满足多会话流、高精度和低成本的链路损伤模拟需求,本文在高良磊研究的单链路粗细粒度循环时延控制的基础上[11],在原始粗粒度时延控制过程中加入二级光纤链路控制,提出一种基于时延量程策略的多会话时延损伤模拟方法,在识别汇聚流进行独立模拟的同时,针对内存占用率高的问题进行有效优化。

1　系统方案总体设计

本文所提出的链路损伤处理过程,主要创新点在于突破传统专用设备的模拟链路数量限制和大幅减少对于内存资源的占用。文献[11]的研究中使用光纤链路与内存缓冲对单会话链路进行时延损伤控制,达到微秒级精度的秒级时延模拟,但其仅仅实现了单条会话流固定时延的损伤模拟,与传统专用设备一样,存在会话流数量严重受限的问题。本文在其研究基础上进一步优化链路结构,采用长短光纤双循环链路提高模拟量程,同时提出基于时延量程策略的队列合并方案,大幅减少内存资源占用,完成多会话时延损伤模拟。

针对链路损伤,主系统首先识别出对应的流量信息,随后每条数据流进入两级粒度的时延模拟控制,长短光纤循环控制粗粒度的时延精度,内存缓冲队列控制细粒度的时延精度。系统总体流程如图 1 所示。

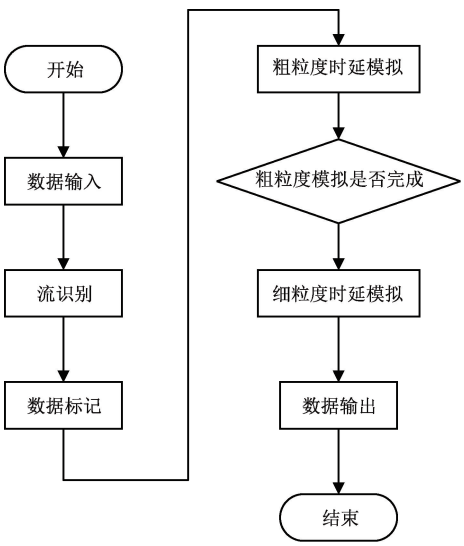

图 1　方案整体流程图

数据源由测试端口输入后,开始经过流识别模块完成汇聚流的流识别检测,过滤筛除无效的背景流量,获取各有效数据流所需要的目标模拟时延值。

后续进行数据标记,为初始数据添加包括模拟时延值与输出时间戳等一系列信息标记,记作元数据,便于在时延损伤模拟过程中各阶段的完成判决。

其次进入两个阶段的时延模拟,分别完成粗细粒度的延时控制。

最后将用于标记的元数据信息从原始数据包中进行剥离,保持数据流量的完整性,再经由输出端口进行输出。

2　时延程策略模块分析设计

2.1　多会话流识别检测模块

1)流识别方式

传统流识别通常采用基于流表策略的方式,针对特定数据流进行特殊的流控行为。例如访问控制列表(access control list,ACL),根据提前设定的条件对接口上的数据流进行处理。但在大规模网络数据流时延损伤模拟场景下,相较于流表识别,采用软件哈希进行流识别检测更快速高效,二者特性对比如表 1 所示。

表 1　特性对比

特性	ACL 流表	软件哈希
原理	基于一系列预定义的规则来识别和控制流量	通过对数据包的关键字段进行哈希计算来识别流
性能	依赖于规则集的大小和复杂性	能够在常数时间内完成查找
可扩展性	低,受限于硬件资源	高,可灵活扩展
可移植性	低,受限于硬件类型	高,适用于通用计算设备

2)哈希表结构

传统流识别中使用数据包的五元组作为流的特征,包含:源网络协议(internet protocol,IP)、目的网络协议、源端口、目的端口与协议类型,属于同一条流的数据包具备相同的五元组。而大规模数据流的标识主要在于介质访问控制(media access control,MAC)地址的差异,于是采用源 MAC 地址与目的 MAC 地址的二元组代替传统五元组作为流标识进行哈希。

另一方面由于传统五元组在获取过程中需要对数据包内容进行深层解析,以获取到 IP 层、网络层的字段内容,具备额外的解析开销。而 MAC 地址位于数据包的以太网帧头部,容易访问。此外由于网络数据规模的激增,目前 IP 层所使用的 IP 地址逐渐从 32 bit 的 IPv4 协议混合成 128 bit 的 IPv6 协议,传统五元组所需的资源开销也会随之增加。

3)哈希函数选型

衡量哈希函数性能的一个重要因素即其哈希函数的优

劣，好的哈希函数能够根据需求均匀的将需要的关键字映射到表中的每一个空闲位置，以降低碰撞的几率。

选型过程中选择使用 Jenkins 哈希与常见的 CRC32 函数进行测试。其中 Jenkins 哈希是基于 Bob Jenkins 的 lookup3 哈希函数，关键特点是能够将输入数据均匀分布到哈希表中，减少碰撞概率。而 CRC32 函数主要用于快速数据完整性校验，核心原理基于循环冗余校验（cyclic redundancy check，CRC），也常用作哈希函数。

2.2　粗粒度时延损伤模块

方案整体采用变光程式光延迟技术，利用光纤作为传输介质模拟物理传输时延。光信号在介质中的传播时延 τ 由式(1)计算得到：

$$\tau = \frac{n_{eff} L_l}{c} \tag{1}$$

其中，n_{eff} 为光传播介质的有效折射率，L_l 为链路长度，c 为真空中的光速。

在文献[11]单光纤链路循环的基础上拆分二级链路进行精度细化控制和小量程拓展。具体连接如图 2 所示。

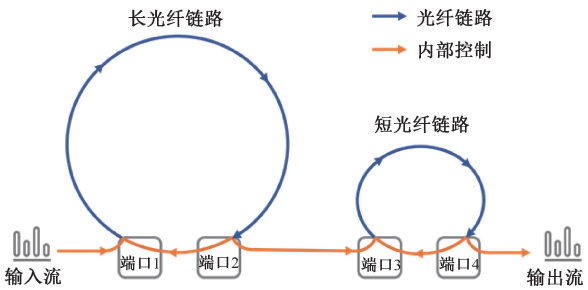

图 2　双链路循环模拟

将整体时延模拟过程划分为两级链路，对链路时延构成进行分析：在将由端口排队时延、处理时延等带来的固有时延作为非循环的背景时延，在进入链路循环之前扣除，总传播时延可表示为：

$$D_{all} = r_l d_l + r_s d_s \tag{2}$$

其中，各自的基时延 d_l 和 d_s 根据光纤长度确定，r_l 和 r_s 为长短光纤链路的循环轮数。

综上，若已知最大循环轮数为 $r_{\max}$，则两级循环链路结构的时延控制范围如表 2 所示。

表 2　两级循环链路结构的时延控制范围

链路	最小时延	最大时延
短光纤链路	d_s	$r_{\max} d_s$
长光纤链路	d_l	$r_{\max} d_l$
双循环链路	d_s	$r_{\max}(d_s + d_l)$

由于短链路光纤单程基时延更小，因此采用双循环光纤链路后可控精度更高。

2.3　细粒度时延损伤模块

延续使用内存缓冲队列对数据流进行精度补偿，提高时延损伤模拟的精度控制。由内存动态补偿的时延部分由文献[11]中定义的精度控制因子 β 确定，β 为内存缓冲部分时延占总体传输时延的权重。

$$T_m = \beta T_{all} \tag{3}$$

通过调整精度控制因子 β 调整最后的精度控制范围。

2.4　基于时延量程策略的多队列合并方案分析与设计

传统软件和物理设备对于多会话流的时延损伤模拟采用流对应的策略，即为每条会话流分配对应的通道进行模拟，在大规模会话流场景下无法适用，极其占用内存资源。因此本文提出基于时延量程维度的队列合并方案对资源占用进行优化。

鉴于多会话流中不同数据流所需模拟的时延值不同，则可以根据特定量程范围进行分组，目标时延值在相邻范围内的数据流在经过粗粒度时延模拟后，经过时延量程判决，进入细粒度时延模拟环节中不同的内存缓冲队列。流程架构如图 3 所示。

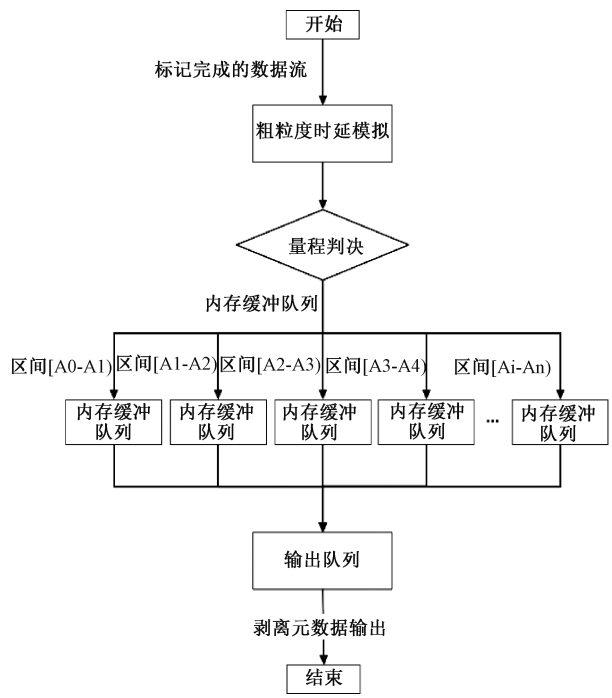

图 3　时延量程维度多队列合并流程图

损伤模拟过程中，原始数据流标记后经过双循环光纤通道完成粗粒度时延控制，再进入细粒度的内存缓冲阶段。首先会经过对数据流所在目标时延的量程判决，划分当前数据流所需进入的队列通道。在划分过程中，将目标时延处于某个量程范围内的数据流分入相同队列进行后续缓冲控制至输出队列，最后剥离元数据输出。

量程区间的极差值即为该区间内的最大控制误差，通过调整极差范围以及端点时延值即可调整控制误差。

2.5　误差分析

由于不同数据流的目标模拟延时不同，在最后的细粒度内存缓冲模块中，数据包会经过原地的内存等待缓冲以

达到目标时延,后续进行输出,对此进行目标为不同时延的数据包在同一个内存缓冲队列中的输出以及误差计算进行分析推导。

分别针对:小时延-大时延、大时延-小时延的不同情形进行分析论证。

如图 4 所示为内存缓冲队列中的数据包排队示意图,*mbuf* 为数据包缓冲区,*data* 为数据包原数据,*app* 为时延模拟过程中添加的信息标定追踪数据。

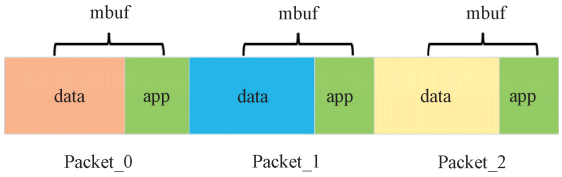

图 4　数据包排队示意图

假设 *packet*_0、*packet*_1、*packet*_2 的内存缓冲时延分别为 a,b,c。

1)小时延-大时延

不妨设 *packet*_0 与 *packet*_1 的输入时间间隔为 n,且 $a<b$。

根据内存缓冲队列的逻辑原则,*packet*_0 在原地等待 a 后,转发进入输出队列,此时 *packet*_1 仍然需要等待 $b-a$ 才达到输出时间,因此误差为 0。

2)大时延-小时延

当 $a>b$ 时,*packet*_0 在原地等待 a 后,转发进入输出队列,而 *packet*_1 已经超时等待 $b-a$。

此时再次存在两种情形:若二者的输入间隔时间 n 大于 $b-a$,则表示 *packet*_1 相对于 *packet*_0 而言,原定需要在 $n+b$ 时刻输出,*packet*_0 输出时的当前时刻相对于输入为 a 时刻。由于 $n+a>b$,故 *packet*_1 在 *packet*_0 完成内存缓冲时并未达到其输出时刻,误差仍然为 0。

反之则表示 *packet*_1 相对于 *packet*_0 而言,原定需要在 $n+b$ 时输出。由于 $n+b<a$,故 *packet*_1 在 *packet*_0 完成内存缓冲之前已经达到其输出时刻,误差为:$a-b-n$。

因此误差范围在:$[0,a-b-n]$。

继续当前情形,大时延数据包在前,小时延数据包在后,将数量增加。此时根据 *packet*_2 的内存缓冲时延模拟大小分类,分为 3 个区间:

$c>a$:在 *packet*_0 经过 a 的延时输出后,*packet*_1 也直接输出,而 *packet*_2 并未达到其输出时刻,故对其无影响,误差值为 0。

$a>c>b$:在 *packet*_0 经过 a 的延时输出后,*packet*_1 也直接输出,而 *packet*_2 已经达到其输出时刻,故存在影响。同样假设其与 *packet*_0 之间的输入间隔为 n,且满足 $n+c<a$。此时对于 *packet*_2 而言,误差值为:$a-c-n$。因此误差范围在:$[0,a-c-n]$。

$a>b>c$:同样的,误差值为:$a-c-n$。因此误差范围在:$[0,a-c-n]$。

总结可得:在自身内存缓冲时延小于前导数据包的目标内存缓冲时延,且加上输入时间间隔依然小于前导数据包的目标内存缓冲时延时,误差最大,即为:前导数据包的目标内存缓冲时延-自身内存缓冲时延-输入时间间隔。将 *packet*_0,*packet*_1 推导至 *packet*_n,仍然符合该结论。因此每个数据包的模拟误差最大不超过与前导数据包的模拟时延差值,控制同一队列中的模拟时延值的极差,即可控制相对误差。

3　实验与分析

为验证本文方案在多会话流链路时延损伤模拟的准确性、完整性,以及内存资源占用方面的优化,设计相关实验,并通过时延模拟软件 NetEm 以及类比传统物理设备模拟进行对比分析。

3.1　实验配置

实验使用基于软件的发包程序,可根据配置,按照指定速率生成 MAC 地址不同的多会话流数据包,搭配物理 10GbE 端口,测量并统计两测试端口间的端到端时延。实验拓扑如图 5 所示。首先发包程序按照配置生成数据包并从网络接口卡 0(network interface card,NIC)发送,同时记录发送时间戳。这些数据包经过待测系统后会被 NIC 1 接收,接收后记录相应的接收时间戳并计算端到端时延。考虑到程序本身在发送和接收过程中存在开销,在进行每次标准时延前,会进行背靠背实验。背靠背实验指待测系统在运行发包程序本身只完成基本的数据包转发功能。两次实验的差值作为最终结果进行统计分析。

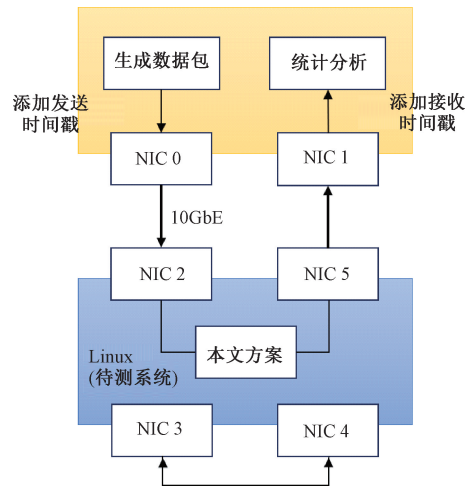

图 5　实验拓扑

原型系统网络 I/O 相关功能基于数据平面开发工具集(data plane development kit,DPDK)实现。使用 DPDK[12-15]后,能够使得高速数据处理性能大大提升。

使用一台 Linux 服务器运行 NetEm 软件和本文方案的原型系统。实验前,需要对服务器进行特定配置实现路

由转发，将图 5 中待测系统 NIC 2 接收到的数据直接转发至 NIC 5 输出。相关步骤包括开启路由转发，并配置 NIC 2 与 NIC 5 的静态路由，保证数据包在端口间的正确转发，最后使用流控(traffic control，TC)命令对 NIC 5 出口数据设置目标时延。

运行本文方案的原型系统时，需要搭建 DPDK 相关环境，包括 DPDK 工程编译、大页内存分配、设置 CPU 隔离、为测试网卡绑定 igb_uio 用户态 I/O 驱动等。完成系列配置后，运行相应测试脚本对本文方案进行测试。服务器配置规格如表 3 所示。

表 3　系统配置

配置项	系统规格
CPU	Intel ® Xeon ® E5-2640 v4@2.4 GHz x4
内存	DDR4@2 667 MHz 32 GB x2
网卡	Intel x710
操作系统	CentOS Linux release 7.9.2009(core)
系统内核版本	6.3.0-1.el7.elrepo.x86_64
DPDK 版本	v20.11

3.2　双循环光纤链路基时延测试

分别对长度为 5 km 与 50 km 的光纤进行不同帧长 10 G 速率下的基时延测试，结果如表 4 和表 5 所示。

设最大循环轮数为 r_{max}，时延量程扩展量记为 D_{Δ}。

$$D_{\Delta} = r_{max} \cdot 24.63\ \mu s \tag{4}$$

相较文献[11]的单链路结构，随着最大循环轮数增加，优化后的双链路结构扩展量即短光纤的可控时延值随之增加。

此外由表 4 和 5 可知，短光纤基时延更小，因此用于时延调节的步长更小，用于时延控制的精度也会更高。

表 4　短光纤基时延测试

测试帧长/Bytes	最小值/μs	最大值/μs	平均值/μs
64	24.48	24.81	24.63
128	24.48	24.82	24.63
256	24.48	24.82	24.63
512	24.48	24.82	24.63
1 024	24.48	24.82	24.63
1 518	24.48	24.82	24.63

表 5　长光纤基时延测试

测试帧长/Bytes	最小值/μs	最大值/μs	平均值/μs
64	245.11	245.42	245.24
128	245.11	245.43	245.24
256	245.11	245.43	245.24
512	245.11	245.43	245.24
1 024	245.11	245.43	245.24
1 518	245.11	245.43	245.24

3.3　多会话流 Jenkins 哈希测试

为测试在大规模会话流场景中的 Jenkins 哈希性能，在哈希表表项与容量比为 0.5 的情况下，分别多次测试 Jenkins 哈希和传统软件使用的 CRC32 哈希在 1 万、10 万、50 万、100 万表项的创建和查表时间。

平均测试结果如图 6 和 7 所示，随着数据量的不断增加，在表创建与插入数据时，传统软件使用的 CRC32 哈希所消耗的开销相较于 Jenkins 哈希逐渐增大。查找时间方面，CRC32 函数所需时间初始更小，随着数据量的增加，表中的数据愈发分布不均，碰撞概率增加，导致后续开销更大。对比之下，Jenkins 函数的查找时间更趋于稳定。

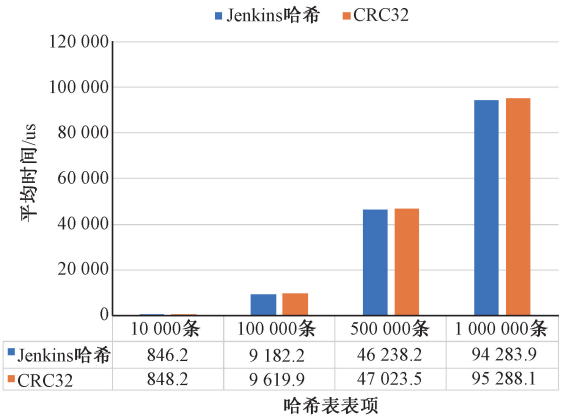

图 6　哈希创建与平均插入时间

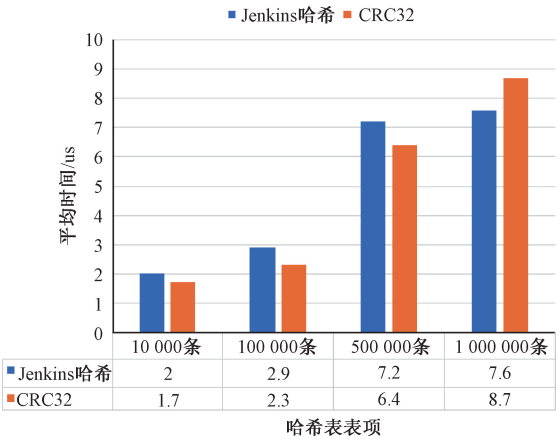

图 7　哈希查找平均时间

3.4　内存资源占用测试

由于传统软件和物理设备对多会话流的时延损伤模拟为单会话流的简易复用，因此先从单流模式下的内存占用进行测试，传输速率取 1 Gbps，测试帧长取 256 字节，内存缓冲补偿系数取 0.8。

测试结果如表 6 所示。

在多会话流场景下，以 10 条独立会话流为例，保持误差最小的情况下测试基于时延量程的多队列动态合并方案与传统队列复用方案的内存占用情况，其中时延程方案将相邻量程的两条队列进行合并，测试结果如表 7 所示。

表 6　双循环链路方法与 NetEm 的内存占用比较

目标时延/μs	NetEm/KB	双循环链路/KB	相对优化率/%
100	18.9	5.6	70.4
300	36.6	8.9	75.7
600	72.3	16.4	77.3
1 200	145.6	25.1	82.8
2 400	276.9	42.6	84.6

表 7　时延量程多队列与传统方案的内存占用比较

目标时延/μs	传统方案/KB	时延程策略/KB	相对优化率/%
100	189	28	85.2
300	366	44.5	87.8
600	723	82	88.7
1 200	1 456	125.5	91.4
2 400	2 769	213	92.3

实验结果中,目标时延相同的情况下,本文提出的基于时延程策略的多队列合并方案相较于传统针对每条流复用一个队列通道的方案,占用内存的优化率至少在 85%以上。且随着独立会话流的增加,传统方案使用的队列数线性增加,而时延量程区间的划分,使得可合并队列增加,从两条队列合并到百、千条队列合并等。因此,随着独立会话流数量的增加,基于时延程策略的队列合并方案优化效果越显著。

由优化率可知,相同内存下,时延程策略二合一队列支持的独立链路数量相较传统方案能够提升 6～10 倍。考虑到流数增加,合并队列数相应增加,到百、千级别合并能够扩展 50～500 倍。此外,相较于专用硬件设备,软件方案可扩展内存更大,结合哈希测试表项结果,本方案支持最高百万级独立链路的时延损伤配置。

3.5　丢包率测试

依次对各个带宽下各个帧长的数据流进行测试,实验结果表明都能零丢包的完成时延损伤模拟并转发,如图 8 所示为其中 1 Gbps 传输速率下帧长 256 bit 的测试结果,过程中仅经过短光纤链路完成时延损伤模拟,系统输入数据包数、短光纤链路接收与转发数据包数、系统输出数据包数四者相一致,无数据包丢失。

```
Input Pkt Num                : 15560586.
Long Forward Pkt Num         : 0.
Long Recv Pkt Num            : 0.
Short Forward Pkt Num        : 15560586.
Short Recv Pkt Num           : 15560586.
Output Pkt Num               : 15560586.
Per Round Average Delay      : 0.000000.
Per Round Maximum Delay      : 0.000000.
```

图 8　丢包率测试

4　结　　论

本文首先介绍了信道链路时延损伤的研究现状,并针对现有物理设备和软件模拟链路数量受限、缺乏灵活性、资源占用高等问题,在文献[11]的时延模拟链路结构基础上进行优化改进,并提出了一种基于时延量程的多会话流链路损伤模拟方法,完成多链路时延损伤控制。最后,从内存占用和模拟准确性角度进行比较和测试验证。实验结果表明,本文方案能够支持百万级别的独立链路配置,会话流数从十数条增加到百万条。此外,相较 NetEm,该方法在各个带宽下降低至少 85%的内存占用,并且随着流数量的增加而进一步提升优化效果。

参考文献

[1] PRATT T, AllNUTT J E. Satellite communications[M]. New York: John Wiley & Sons, 2019.

[2] 蒋加豪. 低轨卫星遥测链路信道模拟关键技术研究[D]. 成都: 电子科技大学, 2022.

[3] 范永鹏. 基于大型低轨卫星星座的网络容量分析[D]. 北京:北京邮电大学,2023.

[4] HEIN G W. Status, perspectives and trends of satellite navigation[J]. Satellite Navigation, 2020, 1(22): 1-12.

[5] CRISP N H, ROBERTS P C E, LIVADIOTTI S, et al. The benefits of very low earth orbit for earth observation missions[J]. Progress in Aerospace Sciences, 2020, 117(8): 1-18.

[6] MLADENOV T, EVANS D, ZELENEVSKIY V. Implementation of a GNU radio-based search and rescue receiver on ESA's OPS-SAT space lab[J]. IEEE Aerospace and Electronic Systems Magazine, 2022, 37(5): 4-12.

[7] 孙金傲,陈茂胜,邹吉炜,等. 微小卫星固存控制系统设计与实现[J]. 电子测量技术,2023,46(10):1-5. DOI: 10.19651/j.cnki.emt.2212251.

[8] 雷鹏. 基于信道模拟的车联网通信系统在环测试方法研究[D]. 长春:吉林大学,2022.

[9] 冯晋文. 基于 SSH 协议的网络损伤调度系统[J]. 工业控制计算机,2020,33(6):103-104,107.

[10] 吕平,董春雷,刘冬培,等. 基于 FPGA 的软件定义流量发生器[J]. 通信学报, 2018, 39(S2):66-71.

[11] 高良磊,曹炳尧,张倩武. 基于链路循环的网络时延模拟方法研究[J]. 工业控制计算机,2023,36(11):42-44,84.

[12] 牟晓玲,董大波,张萍. 基于 DPDK 的自适应数据包解析技术的实现[J]. 数字技术与应用,2023,41(7):77-79.

[13] 朱宜斌. 用 DPDK 和 VPP 优化 5G 网络用户面数据报文转发[J]. 数据通信,2023(4):43-46.

[14] 袁登博,卫红权,朱宇航. 基于 DPDK 的用户态协议栈的设计与实现[J]. 信息工程大学学报,2023,24(1):93-97.

[15] 孙浩然. 基于 DPDK 的 VPN 端到端安全传输技术的研究[D]. 成都:电子科技大学,2023.

作者简介

吴靖,硕士研究生,主要研究方向为通信网络模拟、高性能数据处理。

E-mail:2451109711@qq.com

曹炳尧(通信作者),高级实验师,博士,主要研究方向为高速网络数据分析与处理及网络测试模拟技术。

E-mail:caobingyao@shu.edu.cn