# 高轨卫星数据传输方案优化设计及验证

**史婷婷*，杜文杰，郭志强**

北京计算机技术及应用研究所，北京 100854

**摘要：** 频带资源有限、传输延迟长是高轨卫星和地面之间端到端数据传输的难点。为了解决这些问题，需要对 TCP 实现进行针对性的优化。当前有五种主流的传输方案：双侧 TCP 参数轻度优化方案、双侧 TCP 内核重度优化方案、地端单侧 TCP 内核重度优化方案、重度隧道优化方案和轻量级 TCP-QUIC 代理优化方案。针对高轨卫星的应用场景，通过深入比较分析，最终只有轻量级 TCP-QUIC 代理优化方案能够满足。该方案在在减少连接建立的延迟、提高拥塞控制和流量控制的效率、支持多路复用和连接迁移等方面具有很好的优化效果。利用北京计算机技术及应用研究所建立的原型系统进行的半实物仿真试验对该优化方案进行了验证，经过优化之后，在 150 Mbps 的下行方向上，速率可提升 100 倍以上，10 Mbps 的上行方向上，速率可提升 7 倍以上，完全满足使用需求。轻量级 TCP-QUIC 代理优化方案可作为中国今后的高轨卫星数据传输的很有竞争力的备选标准技术方案，发挥重要的应用价值。

**关键词：** 高轨卫星；数据传输；端到端；传输速率; TCP-QUIC；半实物仿真

**DOI:** 10.57237/j.cst.2024.04.003

# Design and Validation of an Optimized Data Transmission Scheme for High Orbit Satellites

**Tingting Shi*, Wenjie Du, Zhiqiang Guo**

Beijing Institute of Computer Technology and Application, Beijing 100854, China

**Abstract**: The limited frequency resource and low transmission rate are bottleneck of the End-to-End data transmission between high orbit satellites and ground segments. The optimization of the TCP realization is necessary. There are five popular data transmission schemes: double-side TCP lightweight parameter optimization scheme, double-side TCP heavyweight core optimization scheme, heavyweight tunnel optimization scheme and lightweight TCP-QUIC proxy optimization scheme. After intensive analysis, only the lightweight TCP-QUIC proxy optimization scheme can fulfill the user specifications. This scheme has a good optimization effect in reducing the delay of connection establishment, improving the efficiency of congestion control and flow control, and supporting multi-channel multiplexing and connection migration. Based on the prototype system established by Beijing Institute of Computer Technology and Application, hardware-in-the-loop simulations show that, after optimization, the 150Mbps downlink transmission rate can improve by 100 times, while the 10Mbps uplink transmission rate can improve by 7 times. The transmission rates can fully satisfy the user requirements. The optimized scheme in this paper can be a competitive candidate for the scheme of the standard data transmission scheme for future orbit satellites in China.

**Keywords:** High Orbit Satellites; Data Transmission; End-to-End; Transmission Rate; TCP-QUIC; Hardware-in-the-loop Simulation

*通信作者：史婷婷，458066507@qq.com

收稿日期: 2024-10-23；接受日期: 2024-11-28；在线出版日期: 2025-01-09

# 1 引言

随着人造卫星技术的发展，人类已将通信广播卫星送入高轨道，随着航天技术的不断进步和通信需求的日益增长，基于高轨卫星的通信技术得到了快速发展。卫星通信具备覆盖广、稳定性强、容量大等优点，但由于通信卫星多是地球同步卫星，卫星轨道高度达数万公里，同时也因为频段资源和带宽有限，导致信号传输时延过高的问题。因此针对高轨卫星传输场景下，必须对传输方案进行优化设计。

基于卫星的 Internet 数据传输通常和普通的 Internet 一样，需要遵守传输控制协议 TCP（Transmission Control Protocol），近年来国内外学者针对地轨卫星开发了很多 TCP 代理方案[1-7]。高轨卫星信道的时延和误码率不满足 TCP 协议带宽时延乘积小、信道误码率低的理想条件[8, 9]，给 TCP 协议的实现带来了极大的困难。

针对在轨端（高轨卫星）和地面端（地面服务器）之间端到端（End-to-End，简称 E2E）的数据传输问题，本文将对五种主流的传输方案：双侧 TCP 参数轻度优化方案、双侧 TCP 内核重度优化方案、地端单侧 TCP 内核重度优化方案、重度隧道优化方案和轻量级 TCP-QUIC 代理优化方案进行比较分析，并利用半实物仿真进行试验验证。

# 2 优化方案设计

## 2.1 双侧 TCP 参数轻度优化方案

该方案不改变用户主机和服务器的任何软件，仅依靠调节在轨端和地面端现有操作系统允许对 TCP 堆栈的 socket 接口参数进行调节[10]，来优化双向 TCP 传输性能（具体软件设置和协议参数调优参数如图 1）。优点是对现有系统影响小、容易实施；缺点是参数调节空间小、实际验证效果很差，不能高轨卫星传输的满足应用场景要求。出现上述问题的原因是不能从根本上克服信道的高时延低可靠带来的 E2E TCP 拥塞窗口低效率问题，不建议采用此方法进行传输优化。

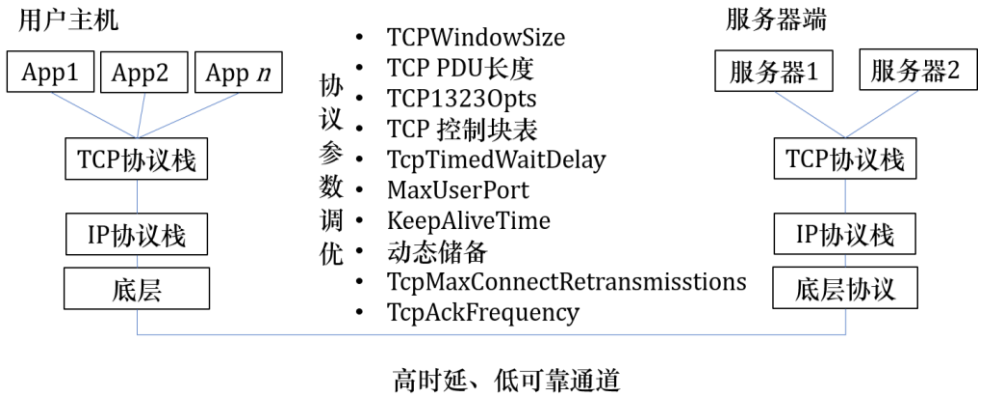


图 1 双侧 TCP 参数轻度优化方案

## 2.2 双侧内核重度优化方案

方案 2 对操作系统内核进行修改编译，优化定制符合场景需求的 TCP 拥塞机制，针对于在轨端和地面端双向优化，编译内核，定制如 Cubic 的 betta 和 C 参数、以及其他 TCP 控制机制。实验设计为：用户主机通过优化的 TCP 协议经由高时延低可靠信道传输至经过 TCP 协议优化的服务器端[11-13]（如图 2）。根据测试（如图 3），经过 BBR 算法优化，上行带宽 100 Mbps 实际传输速率能达到 20~30 Mbps，下行带宽 10 Mbps 实际传输速率能达到 8 Mbps。

该方案优点是 TCP 改进技术较成熟，缺点是看起来很美好，但是，在轨端如果是 Windows 操作系统，由于该操作系统是闭源软件，不开放 TCP 内核修改窗口，实现难度极大；在轨端如果是 Linux 操作系统，虽然可以进行修改，但对在轨端操作系统内核的修改可能会引起“蝴蝶效应”，对其他场景的 TCP 通信服务存在无法预料的风险，地面难以挽救，严重牺牲了系统的可靠性。因此在高轨卫星传输的应用场景下，也不建议采用此方法进行传输优化。


图 2 双侧内核重度优化方案


图 3 双侧内核重度优化方案模拟测试结果

## 2.3 单侧内核重度优化方案

方案 3 和方案 2 的区别是绕开对在轨端操作系统内核驱动的修改，只靠地面端深度优化 TCP 堆栈，修改 TCP 拥塞控制，提高上传速率，编译内核，定制如 Cubic 的 betta 和 C 参数、以及其他 TCP 控制机制；修改 ACK 反馈机制，提高下载速率（如图 4）。该方案优点是避开了对卫星上操作系统的改动，地面可以选择 Linux 系统。但即使经过优化，单侧优化的明显不及双侧优化。上行带宽 100 Mbps 实际传输速率能达到 4 Mbps，下行带宽 10 Mbps 实际传输速率能达到 6 Mbps。这样的传输速率在很多实际应用场景中是不够的。

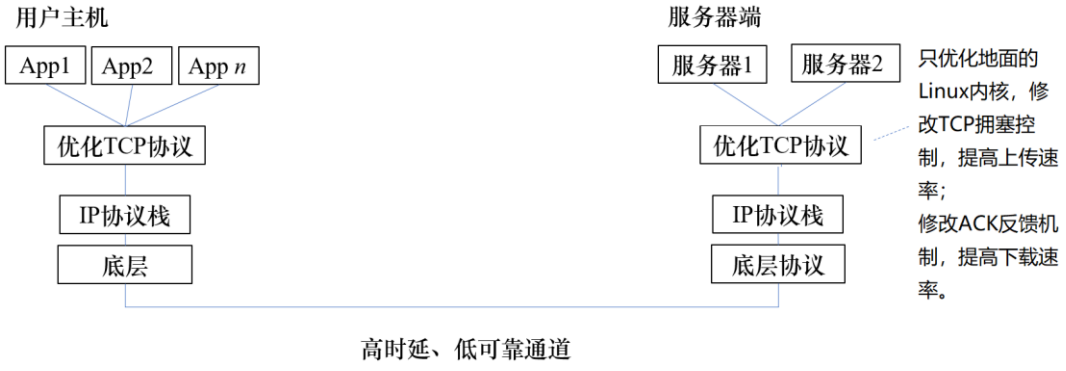


图 4 单侧内核重度优化方案

## 2.4 重度 UDP/FEC 隧道优化方案

方案 4 设计思想是两侧分别连接到设计的 VPN（Virtual Private Network，虚拟专用网络）软件或硬件节点，节点间建立 FEC（Forward Error Correction，前向纠错）/UDP（User Datagram Protocol，用户数据报

协议）隧道透传两侧的 E2E 流[14, 15]。其优点是不改动两侧主机原有软件，VPN 节点可以软件或硬件实现；缺点是隧道不能改变 E2E 的时延，无力改变终端 TCP 拥塞窗口机制导致的传输瓶颈；此外，方案相对复杂。实验效果为实际传输速率远不能达标，不能满足课题要求。

对于重度隧道优化方案，在地面端和在轨端之间建立基于 UDP/FEC 的 VPN 隧道。原有的 TCP 业务数据通过该隧道透传隧道。实验设计为：Linux 主机 Client（客户机）运行 FTP（File Transfer Protocol，文件传送协议）客户端通过运行的 VPN 隧道将数据传输到 Linux 主机 Server 运行的 FTP 服务器。实验结果为上行带宽 100 Mbps 实际传输速率能达到 301Kbps，下行带宽 10 Mbps 实际传输速率能达到 214Kbps。该方案的传输速率尚不如方案 3。实验结果表明，尽管 FEC 能够降低分组丢失，UDP 本身没有拥塞限制，但是，端到端的 TCP 流自身的拥塞控制机制依然受到大往返时延的约束，无法充分利用信道带宽。上行带宽跟下行带宽链路的实际传输速率提高不明显。

## 2.5 轻量级 TCP-QUIC 代理方案

方案 5 的设计思想是在地面端和在轨端之间建立 QUIC（快速）/UDP 的传输通道；将原有的地面端和在轨端的长 TCP 会话打断成两个短 TCP [16]，分别连接并终结在 QUIC 传输通道的两个本地端。相比于其他方案，本方案可以特别显著提升传输速率，且不需要更改两侧操作系统任何底层驱动、协议堆栈和接口，对上层业务透明，突破原有优化框架和现有 TCP 优化机理。本方案的轻量级代理节点就是个普通软件，可支持不同操作系统，安装/配置/管理灵活高效。

综合以上改进方面，TCP 会话本地终结，彻底解决 TCP 窗口瓶颈问题，中间 QUIC/UDP 通道优化高速输送业务数据缺点是应用系统的监听端口要事先确知 TCP 代理相关信息，但这一点可以通过优化设计得以解决。综合考虑，该方案为最优解决方案。


图 5 轻量级 TCP/UDP 代理方案

# 3 轻量级 TCP-QUIC 代理优化方案场景试验结果

针对轻量级 TCP-QUIC 代理优化方案，北京计算机技术及应用研究所开发了一个网络 TCP 传输性能优化系统的原型系统，左侧的 FTP 客户端和优化系统（客户端）部署在一个虚拟机中，右侧的 FTP 服务器和优化系统（服务器端）部署在另外一个虚拟机中，通过设置两个虚拟机之间传输链路参数分别为：从客户端到服务器端链路带宽为 150Mbps，时延为 270ms 或 400ms，误码率为 $10^{-6}$。从服务器端到客户端的链路带宽为 10Mbps，时延为 270ms 或 400ms，误码率为 $10^{-6}$。


图 6 方案 5 实验床设置

半实物实验部署如上图所示，包括两台物理主机 A 和 B，A 中运行 FTP 客户端和 TCP 传输优化客户端软件（原型），二者通过本机网络互联，形成本地 TCP 会话；B 中运行 FTP 服务器和 TCP 传输优化服务器软件（原型），二者也通过本机内网络互联，形成本地 TCP 会话；主机 A 和 B 的传输优化软件通过物理 IP 网络互联，建立高速 QUIC 传输通道。以上实验网络中，主机内网络无限速、无丢包和无时延，而物理网卡需要进行限制以模拟卫星链路。

为了能比较真实的模拟卫星链路，北京计算机技术及应用研究所需要分别对 AB 两端在各自的接收方向上设置时延和丢包，以模拟丢包和时延是发生在卫星链路而不是本地网络；还需要分别对 AP 两端的物理网卡的数据发送置带宽限制，以免抢占额外的卫星链路带宽。

需要说明的是，按照文件传输方向的不同，具体实验参数设置略有不同，具体设置如下。

1. 当测试 FTP 客户端向 FTP 服务器传输文件时，对物理主机 A 的网卡的发送设置限速 150 Mbps 而接收不限速，对该 NIC 发送分组不设置时延和丢包但对接收分组设置（单向）时延 270 或 400 ms 和丢失率 0.2%（考虑主要是 ACK 小分组），同时，我们对物理主机 B 的网卡（NIC）的发送设置限速 10 Mbps 而接收不限速，对该 NIC 发送分组不设置时延和丢包但对接收分组设置（单向）时延 270 ms 和丢失率 1.2%（考虑数据分组按 MTU1500 字节计算）

在以上设置下，我们分别上传 512MB 和 1024MB 的文件各 4 次，并记录实验结果。

2. 当测试 FTP 客户端从 FTP 服务器下载文件时，我们对物理主机 A 的网卡 NICA 的发送设置限速 150 Mbps 而接收不限速，对 NICA 发送分组不设置时延和丢包但对接收分组设置（单向）时延 270 或 400 ms 和丢失率 1.2%（考虑数据分组按 MTU1500 字节计算），同时，我们对物理主机 B 的网卡 NICB 的发送设置限速 10 Mbps 而接收不限速，对 NICB 发送分组不设置时延和丢包但对接收分组设置（单向）时延 270 ms 和丢失率 0.2%（考虑主要是 ACK 小分组）

在上述配置要求下，北京计算机技术及应用研究所设计了以下四种实验场景，并记录实验结果。

场景 1，物理链路设置单向时延 270 ms，上行传输数据，由 FTP 客户端发送文件，FTP 服务器接收数据，传输过程中实验数据如下。

表 1 下行传输过程中的业务传输速率（场景 1）

| 文件大小 | 优化后的速率（Mbps） | | | | 优化前的速率（Mbps） |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 512 MB | 69 | 68 | 67 | 68 | ≈0.6 |
| 1024MB | 68 | 67 | 68 | 68 | ≈0.6 |

表 2 下行传输过程中的文件传输耗时（场景 1）

| 文件大小 | 优化后耗时（s） | | | | 优化前耗时 |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 512 MB | 65 | 62 | 64 | 63 | ≈2 小时 |
| 1024 MB | 122 | 124 | 123 | 123 | ≈4 小时 |

根据表 1 和表 2 可见，优化后的下行业务传输速率得到特别显著的提升，文件传输完成时间显著减小！

场景 2，物理链路设置单向时延 270 ms，上行传输数据，由 FTP 服务器发送文件，FTP 客户端接收数据，传输过程中实验数据如下。

表 3 上行传输过程中的业务传输速率（场景 2）

| 文件大小 | 优化后的速率（Mbps） | | | | 优化前的速率（Mbps） |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 8.6 MB | 7.84 | 7.92 | 8.24 | 7.84 | ≈1.1 |
| 74 MB | 8.32 | 8.32 | 8.32 | 8.32 | ≈1.1 |

表 4 上行传输过程中的文件传输耗时（场景 2）

| 文件大小 | 优化后耗时（s） | | | | 优化前耗时 |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 8.6 MB | 8 | 8 | 8 | 8 | 64 秒 |
| 74 MB | 69 | 68 | 68 | 68 | 590 秒 |

由表 3 和表 4 可见，优化后的上行业务数据传输速率得到特别显著的提升，文件传输完成时间显著减小！

此外，根据实验过程中监视窗口的表现，经过优化后，FTP 客户端在持续上传文件时，发送端的 CPU（Intel8550U, 1.8GHz）占用率由空闲时的 5%上升到 33%，而发送端采用 CPU（Intel i9-9900k, 3.6GHz）时，占用率由空闲时的 1%上升到 10%，观测到的 150 Mbps 带宽的链路上数据速率大约在 115 Mbps 左右。

FTP 客户端在持续下载文件时，接收机的 CPU（Intel8550U, 1.8GHz）占用率由空闲时的 5%上升到平均 12%，而采用 CPU（Intel i9-9900k, 3.6GHz）时，占用率由空闲时的 1%上升到 5%左右，观测到的 10 Mbps 带宽的链路上数据速率大约在 9.3 Mbps 左右。

场景 3，物理链路设置单向时延 400 ms，上行传输数据，由 FTP 客户端发送文件，FTP 服务器接收数据，传输过程中实验数据如下。

表 5 下行传输过程中的业务传输速率（场景 3）

| 文件大小 | 优化后的速率（Mbps） | | | | 优化前的速率（Mbps） |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 512 MB | 47 | 44 | 45 | 50 | ≈0.42 |
| 1024 MB | 45 | 45 | 46 | 48 | ≈0.42 |

表 6 下行传输过程中的文件传输耗时（场景 3）

| 文件大小 | 优化后耗时（s） | | | | 优化前耗时 |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 512 MB | 91 | 93 | 92 | 89 | ≈2 小时 50 分 |
| 1024 MB | 180 | 180 | 179 | 178 | ≈5 小时 50 分 |

由表 5 和表 6 可见，随着 RTT 的增大（到 800ms），优化传输速率有所降低，但下行业务数据速率仍达到 44Mbps 以上；对比优化前后，

优化后的下行业务传输速率依旧有显著提升，文件传输完成时间显著减小！

场景 4，物理链路设置单向时延 400 ms，上行传输数据，由 FTP 服务器发送文件，FTP 客户端接收数据，传输过程中实验数据如下。

表 7 上行传输过程中的业务传输速率（场景 4）

| 文件大小 | 优化后的速率（Mbps） | | | | 优化前的速率（Mbps） |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 8.6 MB | 7.06 | 7.10 | 7.08 | 7.20 | 0.72 |
| 74 MB | 7.64 | 7.60 | 7.70 | 7.68 | 0.72 |

表 8 上行传输过程中的文件传输耗时（场景 4）

| 文件大小 | 优化后耗时（s） | | | | 优化前耗时 |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 |
| 8.6 MB | 11 | 8 | 10 | 9 | 89 秒 |
| 74 MB | 74 | 75 | 71 | 72 | 760 秒 |

由表 7 和表 8 可见，随着 RTT 的增大（到 800 ms），优化传输速率有所降低，但优化后的上行业务数据传输速率依旧得到显著的提升，文件传输完成时间显著减小！

实验结果显示，按照我们提出的思路设计的 TCP 传输优化系统（原型系统）能够大幅度提升上行和下行方向上的业务数据传输速率，效果非常显著，在 150Mbps 的下行方向上，速率仍可提升 100 倍以上，10Mbps 的上行方向上，速率可提升 7 倍以上，远超使用需求。即使往返时延从 540ms 升高到 800ms，优化传输结果依旧非常显著。此外，该技术方案兼容原有业务接口，无须改动网络协议堆栈和操作系统底层代码，对业务透明，符合技术要求书规定的基本设计要求。

## 4 各方案技术路线和实验结果总结

根据前文的方案比较和试验验证结果，总结出各方案技术路线和实验结果一览表，如表 9 所示。

表 9 各方案技术路线和实验结果一览表

| 方案序号 | 方案名称 | 技术路线 | 实验结果 | 基本结论 |
|---|---|---|---|---|
| 1 | 双侧 TCP 参数轻度优化方案 | 只靠调节在轨端和地面端现有操作系统允许调节的 TCP 堆栈的 socket 接口参数，来优化双向 TCP 传输性能 | 轻量级优化方案优化效果非常有限，远远不能满足设计要求的传输速率。 | 不可行 |
| 2 | 双侧 TCP 内核重度优化方案 | 在地面端和在轨端双侧，在操作系统内核实现能兼容 TCP 接口的新传输控制协议，替换传统的 TCP 传输机制。 | 在轨端若采取 Windows 内核逻辑复杂、技术文档不开放，实际设计开发难度极大；在轨端若采用 Linux 内核，上行带宽 100 Mbps 实际传输速率能达到 20~30 Mbps，但修改操作系统带来的未知风险无法控制。 | 不可行 |
| 3 | 地端单侧 TCP 内核重度优化方案 | 只深度优化地面端单侧的 TCP 堆栈，不改动天端堆栈，包括：修改/替换 TCP 拥塞控制算法，提高地-天的传输速率；修改地端的 ACK 反馈机制，提高天-地的传输速率 | 容易实施：地端深度优化、天端无需修改。但是实验结果表明：10Mbps 链路上的实际传输速率一定提升，能够达到约 4Mbps 左右，不能满足设计要求；100Mbps 链路上实现的传输速率约 6Mbps 左右，不能达到要求。 | 不可行 |
| 4 | 重度隧道优化方案 | 在地面端和在轨端之间建立基于 UDP/FEC 的 VPN 隧道。原有的 TCP 业务数据通过该隧道透传隧道。 | 尽管 FEC 能够降低分组丢失，UDP 本身没有拥塞限制，但是，端到端的 TCP 流自身的拥塞控制机制依然受到大 RTT 的约束，无法充分利用信道带宽。 实验结果表明：10/100Mbps 链路的实际传输速率提高均不足 1Mbps，远不能达到要求。 | 不可行 |
| 5 | 轻量级 TCP-QUIC 代理优化方案 | 在地面端和在轨端之间建立 QUIC/UDP 的传输通道；将原有的地面端和在轨端的长 TCP 会话打断成两个短 TCP，分别连接并终结在 QUIC 传输通道的两个本地端。 | 优势：TCP 会话被终结在本地，RTT 小、无窗口瓶颈；QUIC/UDP 传输通道专门针对高延迟/带宽/丢失率进行优化设计，因此能够充分利用信道带宽。<br>实验结果表明：10Mbps 带宽实际可达 8.5Mbps；100Mbps 的实际可高达 72.5Mbps。 | 可行，效果理想 |

## 5 结论

本文针对在轨端（高轨卫星）和地面端（地面服务器）之间的数据传输问题，对五种主流的传输方案：双侧 TCP 参数轻度优化方案、双侧 TCP 内核重度优化方案、地端单侧 TCP 内核重度优化方案、重度隧道优化方案和轻量级 TCP-QUIC 代理优化方案进行比较分析，最终只有轻量级 TCP-QUIC 代理优化方案能够满足使用要求。利用北京计算机技术及应用研究所建立的原型系统进行的半实物仿真试验结果表明，经过优化之后，在 150Mbps 的下行方向上，速率可提升 100 倍以上，10 Mbps 的上行方向上，速率可提升 7 倍以上，远超使用需求。因此，轻量级 TCP-QUIC 代理优化方案可作为中国今后的高轨卫星数据传输的很有竞争力的备选标准技术方案，发挥重要的应用价值。

## 参考文献


[1] Carlo Caini, Rosario Firrincieli, Daniele Lacamera. PEPsal: a Performance Enhancing Proxy for TCP satellite connections [C], Proc. of IEEE VTC'06 spring, Melbourne, May 2006, 22(8): B9-B16.

[2] 曾 斌, 李之棠, 徐帆江. 面向卫星网络的 TCP 代理 [J], 软件学报, 2007, 18(7): 1695-1704.

[3] D. Velenis, D. Kalogeras, B. Maglaris. SaTPEP: A TCP Performance Enhancing Proxy for Satellite Links Published in NETWORKING 19 May 2002 Computer Science, Engineering: 1233-1238.

[4] 王辉, 王凌云, 吴震. 面向卫星网络的 TCP 传输性能的研究 [J], 通信技术, 2008, 41(9): 145-149.

[5] 侯晓谦. 适用于卫星网路的 T CP Vegas 算法的改进研究 [J], 科技视界, 2015, 1(84): 119.

[6] 林晓拔. 卫星互联网 TC P 拥塞控制算法 [J], 计算机工程与设计, 2018, 4(39): 94.

[7] X. Cao and X. Zhang. SaTCP: Link-layer informed TCP adaptation for highly dynamic LEO satellite networks [C], IEEE INFOCOM, Hoboken, New Jersey, USA, 2023.

[8] W. Stevens, TCP/IP Illustrated, Volume I: The Protocols [M]. 北京: 清华大学出版社, 1994.

[9] Sibo Pan, Ruifeng Gao, Yingdong Hu, Ye Li. TCP Performance in Satellite Backhauling for Maritime Communications: A ns-3 Study [C], 14th International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, November 2022: 877-882.

[10] Red Hat Enterprise. Chapter 31. Tuning the network performance, in Monitoring and managing system status and performance: [M], 2024: 244-285.

[11] Jerry Chu. Tuning TCP Parameters for the 21st Century [C], 75th Internet Engineering Task Force Confernce, Stockholm, July 2009.

[12] Yuchung Cheng, Neal Cardwell, Making Linux TCP Fast [J], Netdev Conference, 2023.

[13] ExtraHop Network Inc. Optimizing TCP: Nagle's Algorithm and Beyond [R], 2016. https://assets.extrahop.com/whitepapers/TCP-Optimization-Guide-by-ExtraHop.pdf

[14] 林利祥, 刘旭东, 刘少腾, 徐跃东. 前向纠错编码在网络传输协议中的应用综述 [J], 计算机科学, 2022, 49(2): 292-303.

[15] Esma Yildirim, Dengpan Yin, Tevfik Kosar, Balancing TCP Buffer vs Parallel Streams in Application Level Throughput Optimization [C], DADC'09, Munich, Germany. June 2009: 21-30.

[16] M. Y. Sanadidi, M. Gerla, J. Stepanek. On-board satellite 'Split TCP' Proxy. IEEE Journal on Selected Areas in Communications, 2004, 22(2): 362 – 370.