计算机研究与发展
Journal of Computer Research and Development
https://crad.ict.ac.cn
63(11): 2925−2938, 2026

SHP-BBR: 一种基于卫星切换预测的 TCP 拥塞控制机制

刘永基　李成龙　黄安文　李　琼　刘洋徐瑞　宋　鑫
（军事科学院国防科技创新研究院　北京　100071）
（liuyongji77@163.com）

SHP-BBR: A TCP Congestion Control Mechanism Based on Satellite Handover Prediction

Liu Yongji, Li Chenglong, Huang Anwen, Li Qiong, Liu Yangxurui, and Song Xin
（*National Innovation Institute of Defense Technology*, *Academy of Military Sciences*, *Beijing* 100071）

Abstract　With the rapid development of integrated space–air–ground networks and large-scale constellations, low earth orbit satellite networks (LEO-SN) together with unmanned aerial vehicles (UAV) constitute a hierarchical and heterogeneous aerial access/backhaul backbone, realizing three-dimensional “space–air–ground” complementarity and becoming a research hotspot for 6G. Owing to the high packet-loss rate, long latency, and pronounced dynamics of LEO-SN links, the Transmission Control Protocol (TCP) congestion-control mechanisms designed for terrestrial networks cannot maintain stable and efficient bandwidth utilization. Compared with traditional loss-based and delay-based algorithms, probe-based algorithms represented by BBR (bottleneck bandwidth and round-trip propagation time) are better suited to such complex environments; nevertheless, theoretical analysis and experiments show that, limited by its passive sensing strategy, BBR can suffer a sustained throughput drop during satellite handovers—the so-called “long trough effect”. To address this issue, we propose SHP-BBR, a TCP congestion-control mechanism based on satellite-handover prediction. SHP-BBR leverages historical satellite-handover records and round-trip-time data to build a time-series prediction model that forecasts upcoming handover events. Guided by these predictions, SHP-BBR proactively senses impending link-state changes, obtains accurate network-parameter estimates, and preserves bandwidth utilization. Experimental results demonstrate that SHP-BBR stably eliminates the long trough effect in LEO-SN environments, achieving an average throughput gain of 33%. Moreover, compared with conventional TCP congestion-control algorithms, SHP-BBR delivers efficient, well-balanced throughput and latency under dynamic network conditions.

Key words　low earth orbit satellite networks; TCP congestion control; satellite handover prediction; BBR algorithm; network performance optimization

摘　要　随着“空-天-地”一体化网络和大规模星座的快速发展，低轨卫星网络（low earth orbit satellite network, LEO-SN）与无人机（unmanned aerial vehicle, UAV）共同构成分层异构的空中接入/回传骨干，实现“空-天-地”3 维互补并成为当前 6G 研究热点。由于 LEO-SN 链路的高丢包、长时延和高动态，现有地面网络的 TCP（Transmission Control Protocol）拥塞控制机制难以保持稳定且高效的带宽利用率。相较于依赖丢包和时延反馈的传统算法，以 BBR（bottleneck bandwidth and round-trip propagation time）为代表的探测型 TCP 拥塞控制算法更适用于此类复杂环境；然而，通过理论分析和实验证明，BBR 算法受限于被动感知策略，在卫星链路切换时可能导致持续的带宽下降，即“长波谷效应”。为解决上述问题，提出了一种基于卫星切换预测的 TCP 拥塞控制机制 SHP-BBR，利用历史卫星切换数据和往返时延数据，构建时间

收稿日期：2025-08-27；**修回日期**：2026-05-26
通信作者：黄安文（awhuang@163.com）

序列预测模型,提前预测卫星切换事件。基于预测结果,SHP-BBR 能够提前主动感知链路状态变化,获得准确的网络参数估计,保证网络的带宽利用率。实验结果表明,SHP-BBR 在低轨卫星网络环境中能够稳定消除“长波谷效应”,实现平均 33% 的带宽提升。同时,与传统 TCP 拥塞控制算法相比,SHP-BBR 在动态变化的网络环境中表现出了高效、平衡的吞吐量和时延特性。

关键词 低轨卫星网络;TCP 拥塞控制;卫星切换预测;BBR 算法;网络性能优化

中图法分类号 TP393; TN915.04

DOI: 10.7544/issn1000-1239.202550643 **CSTR:** 32373.14.issn1000-1239.202550643

随着星座规模的迅猛扩张以及航天成本的持续下降,低轨卫星网络(low earth orbit satellite network, LEO-SN)正在成为全球宽带接入和“空-天-地”一体化互联的研究与产业热点[1]。与此同时,无人机由于部署迅速、机动灵活和视距传播条件良好,正被广泛用于灾害救援、应急通信、物流投递及边远地区临时覆盖等场景[2]。在“空-天-地”一体化网络的构想中,无人机处于距地面 100~3 000 m 的“空”域,和低轨卫星共同构成分层异构的空中接入/回传骨干,从而实现“空-天-地”3 维互补。

现有地面网络协议体系中,TCP 是最常用的可靠传输协议。TCP 使用拥塞控制机制在端到端范围内动态调整发送速率,以避免网络拥塞并兼顾传输效率与公平性。如何设计面向 LEO-SN 的 TCP 拥塞控制机制,是实现高效“空-天-地”一体化通信的基础。

LEO-SN 受到特定轨道高度(300~1 500 km)和星座规模(上千颗卫星)的影响,其网络特征与地面网络主要有以下区别:1)高丢包。星间链路和星地链路均受限于动态波束、天气衰减和射频干扰,随机丢包率通常在 10^{-7}~10^{-2} 范围内波动[3],远高于地面网络的丢包率。2)长时延。单跳传播时延可达 10~30 ms,多跳星间转发及队列排队后,端到端时延可达几百毫秒[3],甚至高于部分地面长途骨干链路。3)高动态。低轨卫星的高速移动使网络处于高动态环境。用户终端通常在数十秒到几分钟内经历一次卫星切换。拓扑高频变化直接影响到往返时延(round-trip time, *RTT*)、链路容量以及排队状态。UAV 接入 LEO-SN 时,则更进一步加剧了这种高动态。

LEO-SN 的网络特征对传统 TCP 拥塞控制算法提出了较大挑战:1)基于丢包的算法(如 Reno, BIC, CUBIC)会误将随机无线丢包解释为拥塞信号,导致过度减小拥塞窗口(congestion window, *CWND*),长期处于“拥塞避免-快速重传”震荡,难以充分利用带宽。2)基于时延的算法(如 Vegas, TIMELY)依赖 *RTT* 判断网络中积压排队的数据量。然而 LEO-SN 的 *RTT* 波动既有排队因素,也有卫星轨道变化导致的传播时延漂移,时延-拥塞判别边界模糊,易产生误判。3)基于探测的算法(BBR 算法及其各种变体)通过周期性估算网络状态,建立对网络可承载数据量的认识,控制发送至网络的数据量,相较前 2 种算法,能更好地应对 LEO-SN 的高动态特性。然而,本文针对 BBR 算法开展了 LEO-SN 环境 TCP 传输的模拟实验,发现在卫星链路切换时,如果新链路固有时延显著高于旧链路固有时延,BBR 算法会因为无法及时开始网络状态探测导致持续长时间带宽下降现象。本文将上述现象称为“长波谷效应”,通过理论分析与实验证明对该现象进行深入分析。

为解决“长波谷效应”问题,本文提出一种基于卫星切换预测的 TCP 拥塞控制机制(SHP-BBR),其核心设计思想如下:

1)切换预测模型。利用历史卫星切换数据和历史 *RTT* 数据,结合神经网络模型的时间序列预测方法,提前预测是否即将发生卫星切换事件。

2)链路状态重估。在预测窗口内,持续测量新链路的传播时延与可用带宽,进行快速、准确的新链路网络参数估计。

3)拥塞窗口与发送速率调度。将预测结果与新的网络参数估计反馈至拥塞控制算法,实现拥塞窗口的快速过渡与发送速率调整。

为验证所提出机制的有效性,本文使用开源 LeoEM 模拟器[4]仿真了真实 LEO-SN 通信环境,基于 Linux 环境实现了 SHP-BBR 原型系统。实验结果表明,本文所提出的 SHP-BBR 算法成功消除了“长波谷效应”问题,能够提升约 33% 的传输带宽利用,仅引入约 9% 的时延增长开销。

1 相关工作

1.1 典型 TCP 拥塞控制机制分析

TCP 拥塞控制机制的作用是在网络中出现丢包

或延迟变化时调整发送窗口，以避免因传输数据过载而导致带宽浪费和链路崩溃。其基本原理包括慢启动、拥塞避免、快重传和快恢复等机制，其核心目标是根据实际网络状态动态调整传输速率，平衡网络资源利用率与传输可靠性。

根据调整机制的不同，典型 TCP 拥塞控制算法主要可分为基于丢包、基于时延、基于探测的算法。

基于丢包触发的算法如 NewReno[5]，CUBIC[6]，以丢包作为拥塞判定依据，乘性减小拥塞窗口，对于无线信道中因信号衰减、噪声等引起的误码丢包的识别能力较弱。在 LEO-SN 中，网络高动态变化与丢包率不稳定带来了诸多问题。首先是拥塞误判，低轨卫星链路中，由于天气等外部因素引起的随机丢包率较高，该类算法往往将部分误码丢包误判为拥塞，从而缩小窗口、降低吞吐量。其次，当遇到链路中断或短时带宽下降时，该类算法的拥塞恢复机制响应不足，窗口调整较为滞后，进一步降低传输效率。

针对丢包率较高的无线网络场景，Westwood[7] 通过对 ACK 速率进行滤波，估计可用带宽，并在发生丢包时依据带宽估计值与时延估计值自适应调整拥塞窗口，一定程度上缓解了上述丢包型算法的带宽陡降问题。然而，在 LEO-SN 中，卫星运动和卫星链路切换会导致链路带宽与传播时延频繁变更，Westwood 基于历史 ACK 信息的带宽估计、时延估计容易失效。因此，Westwood 类算法虽然针对随机丢包有一定优化效果，但在 LEO-SN 中仍面临吞吐下降和收敛缓慢等问题。

基于时延反馈的算法如 Vegas[8]，依赖 *RTT* 变化判断网络拥塞情况，过高 *RTT* 会使得算法对网络状态的反馈响应变慢，导致在连接建立初期和网络状态变化时，发送窗口调整滞后。此外，LEO-SN 的高动态导致了时延不稳定，这使得 *RTT* 测量值不可靠，进而出现拥塞误判问题。Vegas 对丢包情况的判断也依赖于 *RTT* 测量，这使得它往往对丢包反应不当而持续维持低的窗口大小，从而影响整体的吞吐率。在不可靠的低轨卫星环境中，较高的丢包率使得此问题更加突出。

基于探测的算法如 PCC[9] 将拥塞控制问题建模为一个以吞吐、时延和丢包等指标构成的效用最大化问题，通过在端侧主动探测不同发送速率并利用在线学习方法选择最优策略，从而摆脱了对传统丢包或 *RTT* 阈值信号的依赖。在 PCC 的基础上，Copa[10] 从控制与优化理论角度进一步发展了端侧优化的拥塞控制思想。Copa 将拥塞控制形式化为吞吐与排队时延之间的权衡问题，通过调节发送速率使网络运行在接近最优操作点的位置。该算法仅依赖端侧 *RTT* 变化进行控制，在多种有线网络环境中能够实现稳定的吞吐与时延的折中。上述端侧优化型算法通常假设网络状态变化相对平滑，且 *RTT* 的变化主要由路由节点排队引起。在 LEO-SN 中，上述假设通常不成立，网络状态快速变化，*RTT* 受到卫星链路切换影响难以反映排队状态。

BBR[11] 算法的拥塞控制过程基于带宽时延积（bandwidth-delay product，*BDP*）模型，通过实时估算瓶颈链路带宽与传播时延，试图使发送速率趋近于链路物理极限，从而达到高吞吐量和低延迟的目标。在 LEO-SN 中，BBR 通过主动测量和速率调节，在稳定链路条件下能够充分利用 LEO-SN 的带宽潜力。同时，利用 *RTT* 最小值估算，理论上可以降低缓冲区数据积压，从而减小排队延迟。BBR 对于丢包不敏感，使其不受 LEO-SN 的高丢包影响；不依赖时延作为节点排队估计，使网络拥塞与 LEO-SN 的频繁时延变化解耦；基于 *BDP* 模型的探测机制，可以持续更新网络状态估计，使其能一定程度上应对 LEO-SN 的动态性。相较于以上各类算法，BBR 更适应于 LEO-SN。

1.2 LEO-SN 环境的 TCP 拥塞控制实验

1.1 节从原理上分析了各类型的 TCP 拥塞控制算法在 LEO-SN 环境面临的挑战，此外，有很多相关研究通过实验手段对各算法的性能表现进行了对比，为本研究选择一个更高效的基线算法进行优化提供了依据。

Cao 等人[4] 设计实现了 LEO-SN 模拟器 LeoEM。基于该模拟器，他们构建了 Starlink Shell 1 LEO-SN，选取了多个源-目的对（例如：圣地亚哥至纽约、西雅图至上海等）进行 TCP 传输实验，采用不同的链路类型（激光链路 ISL 与弯管链路 BP），模拟了 CUBIC 算法与 BBR 算法在 LEO-SN 下的不同表现。在多个情况下，CUBIC 的吞吐量仅为该链路容量的 20%~30%，BBR 的吞吐量则在 90% 左右，二者表现差异显著。此外，CUBIC 在网络动态变化频繁时容易撤回传输窗口，导致明显的延迟增大，其 *RTT* 在高动态条件下波动较大。BBR 通过延迟反馈进行自适应调整，尽管在某些 ISL 路径上也表现出因延迟变动导致的瞬时吞吐低，但整体仍表现平稳。

王子涵等人[12] 通过设置一个包含 2 台发送端和 1 台接收端的网络拓扑，其中一个发送端使用 Vegas，另一个发送端使用 BBR。在 Linux 系统环境中，使用 iperf 软件进行网络吞吐量测试。实验结果显示，在无误码丢包的条件下，Vegas 在链路条件变化时，

吞吐量持续降低，几乎无法充分利用网络带宽。BBR 在经历了一个短时间的低吞吐量后逐渐恢复，但频繁进入时延探测阶段影响了数据传输的稳定性。当设置误码率为 10^{-7} 时，Vegas 几乎没有吞吐量，而 BBR 的性能稳定，能够有效应对传输中的丢包情况。

Barbosa 等人[13]使用 Hypatia 仿真器，模拟基于 Starlink 卫星星座的多跳激光链路，分别对 NewReno, CUBIC, Vegas, BBR 算法进行了不同距离下的对比实验。实验结果显示，NewReno 与 CUBIC 在吞吐量上相对较高，但延迟较大。Vegas 虽然延迟最小，但吞吐量最低，且在动态变化下不稳定。BBR 在吞吐量和延迟间取得了较好的平衡，能较好适应路径变化。

综合来看，各算法均不能达到其在地面网络上的性能表现。其中，基于探测的 BBR 算法在吞吐量和时延表现上，较其他算法更具有优势。

1.3 面向 LEO-SN 优化的 TCP 拥塞控制

为适应 LEO-SN 特性，近年来学术界提出了多种优化的 TCP 拥塞控制机制，以期提升数据传输性能。

AdaBoost-TCP[14]采用集成学习算法，对丢包进行分类，通过分析 *RTT* 和相邻确认应答（acknowledgment, ACK）的时间间隔，判断丢包是否由网络拥塞引起。该算法利用显式拥塞通知（explicit congestion notification, ECN）标志将信息反馈给发送端，使其能够动态调整传输速率。RCC-TCP[15]引入了长短期记忆网络来预测网络拥塞程度，同样旨在减少因非拥塞丢包引起的性能下降。TCP-BCLP[16]结合差分自回归移动平均模型与卡尔曼滤波，估计端到端的可用带宽与拥塞等级，基于量化的拥塞等级实现拥塞控制。

这些算法虽在整体上提升了传输性能，但在卫星切换后受网络参数变化影响，性能都存在下降问题。为应对卫星切换事件的影响，LDM-Satellite[17]引入了漂移检测模型，通过改进样本的分类方法，进一步提高了丢包判别的准确率，但仍然对某些类型的数据漂移（如往返漂移、突变点漂移）处理不佳。此外，基于丢包判别的拥塞控制机制介入时间均在网络拥塞发生后，存在拥塞控制不及时的问题。

在 LEO-SN 中，卫星通常处于高速飞行状态，地面终端接入卫星的链路是持续动态变化的，链路时延随着接入卫星距离而动态变化，当距离大于一定阈值则会触发接入卫星链路切换。Deep-BBR[18]通过结合长短期记忆网络和图卷积网络，实现了带宽利用率的准确预测，并据此动态调整 BBR 中的发送速率增益（pacing gain）值，以匹配卫星网络的当前状况。该方法重点优化了卫星移动过程中的响应速度，但在面对卫星切换时仍存在带宽下降问题。DDTCP[12]针对 BBR 算法进行了优化，通过监测 ACK 到达时间并计算 *RTT* 的变化趋势，判断卫星切换的发生，从而优化状态探测的精准度。这种方法在卫星切换导致传输降速之后生效，反应速度不够及时。

SaTCP[4]通过来自卫星地面站的主动卫星切换通知，避免了 CUBIC 算法的拥塞误判问题。类似地，基于 ECN 机制，研究人员也提出了多种适应性算法，通过在路由设备中标记确认数据包，从而快速反馈网络拥塞状态[19-22]。这类方法能够引入更直接的拥塞通知机制，然而实施起来需要非端侧的其他设备支持，或更改现有网络路由器的处理规则，从而增加了复杂性与成本。

LeoCC[23]将 LEO-SN 建模为由连接重配置事件分隔的离散稳态区间，通过持续检测重配置事件、清空历史数据、重新估计网络参数、调整速率，设计了一个反应式的、事后自适应的新拥塞控制算法。LeoCC 虽然取得了整体上的带宽与时延的提升，但其引入了常驻的高频重配置探测流（原文设置为 10 ms 间隔），增大了网络负载。同时，LeoCC 引入了更大的内存存储开销，突破了主流 Linux 内核中 ICSK_CA_PRIV_SIZE 的限制，需要修改内核宏定义并重编译，在主线/发行版内核默认配置下无法以模块形式直接部署，增加了移植与部署门槛。

尽管当前的研究在针对 LEO-SN 的优化方面已取得了一定进展，但整体上，涉及到卫星切换事件时的表现仍未达到理想水平。考虑到卫星切换事件在 LEO-SN 中频繁发生，本研究进一步分析 LEO-SN 的高动态性，并设计仅在端侧工作的更为高效的 TCP 拥塞控制算法，而不需要其他设备的支持，提升 LEO-SN 的整体性能。

2 研究动机

BBR 算法受益于其对链路状态的探测机制，在 LEO-SN 环境下相较于其他拥塞控制算法更具优势。因此，本文基于 BBR 算法开展进一步优化研究。

2.1 BBR 算法分析

BBR 算法通过对链路固有时延估计（round-trip propagation, *RTprop*）和链路瓶颈带宽估计（bottleneck bandwidth, *BtlBw*）两个核心变量，判断网络链路的可承载数据量，即 *BDP*:

$$BDP = RTprop \times BtlBw \text{。} \tag{1}$$

BBR 算法通过控制拥塞窗口大小和发送速率，

保持在网传输数据量与 *BDP* 相等，从而实现对网络拥塞的规避。

BBR 算法实现拥塞控制的核心过程如算法 1 所示。首先是状态初始化（行①~③），将拥塞窗口和发送速率等初始化为默认值，并进入 STARTUP 阶段。这为算法提供了一个起始点，使其能够根据当前网络状态进行动态调整。

下面重点分析 BBR 算法中的 *RTprop* 更新策略。由于测量时延通常不会低于链路固有时延，BBR 算法通过比较测量到的 *RTT* 与 *RTprop*，取较小值更新 *RTprop*（行⑤~⑨）。这个步骤确保了 *RTprop* 能够反映链路的真实时延情况，但当 LEO-SN 经历频繁的卫星切换事件时，新的链路固有时延往往会有所变化。当新固有时延显著高于旧固有时延时，BBR 算法测量到的 *RTT* 总是高于 *RTprop*，因此无法对 *RTprop* 进行更新。此时 *RTprop* 较真实固有时延偏小，进一步导致 *BDP* 计算值偏小，BBR 算法为保证在网传输数据与 *BDP* 之间保持平衡，会减小拥塞窗口和发送速率。这意味着 BBR 算法在卫星切换事件后，可能导致吞吐量持续下降，无法充分利用网络带宽资源。

算法 1. BBR 拥塞控制核心算法。

输入：*RTT* 测量值 *measuredRTT*；

输出：拥塞窗口 *CWND*，发送速率 *pacing_rate*。

① 初始化 *CWND*；
② 初始化 *pacing_rate*；
③ 初始化 *RTprop_update_time*；
④ while true do
⑤ *measuredRTT*=*Measure_RTT*();
⑥ if *measuredRTT*<*RTprop* then
⑦ *RTprop*=*measuredRTT*;
⑧ *RTprop_update_time*=*now*();
⑨ end if
⑩ *BtlBw*=*Estimate_BtlBw*();
⑪ *BDP*=*RTprop*×*BtlBw*;
⑫ *CWND*=*Update_CWND*(*BDP*);
⑬ *pacing_rate*=*CWND*/*RTprop*;
⑭ if *now*()−*RTprop_update_time*>*threshold* then
⑮ *RTprop_update_time*=*now*();
/*进入 *RTT* 探测阶段*/
⑯ *CWND*=4;
⑰ *measuredRTT*=*Measure_RTT*();
⑱ *RTprop*=*measuredRTT*;
⑲ end if
⑳ end while

本文建立如下数学模型说明吞吐量下降问题。设 t_s 为卫星切换时刻，切换前后的链路固有时延分别为 $RTT_{\text{prop}}^{\text{old}}$ 和 $RTT_{\text{prop}}^{\text{new}}$，且满足 $RTT_{\text{prop}}^{\text{new}} > RTT_{\text{prop}}^{\text{old}}$。根据 BBR 算法的 *RTprop* 更新机制（行⑤~⑨）：

$$RTprop(t)=\begin{cases} RTT_{\text{prop}}^{\text{old}}, & t\in[t_s, t_s+T_w], \\ RTT_{\text{prop}}^{\text{new}}, & t>t_s+T_w, \end{cases} \tag{2}$$

其中 T_w 为时间窗口长度（T_w 典型值为 10 s）。在 $[t_s, t_s+T_w]$ 时间段内，BBR 算法计算的 *BDP* 值为

$$BDP_{\text{calc}}(t)=RTprop(t)\times BtlBw(t)=RTT_{\text{prop}}^{\text{old}}\times BtlBw(t)。 \tag{3}$$

而实际链路的 *BDP* 为

$$BDP_{\text{real}}(t)=RTT_{\text{prop}}^{\text{new}}\times BtlBw(t)。 \tag{4}$$

定义带宽利用率偏差 η 为

$$\eta=\frac{BDP_{\text{calc}}(t)}{BDP_{\text{real}}(t)}=\frac{RTT_{\text{prop}}^{\text{old}}}{RTT_{\text{prop}}^{\text{new}}}<1, \tag{5}$$

η<1 表明卫星切换前后链路时延差会导致带宽利用率的下降，且时延差越大，对吞吐量的影响越明显。

BBR 算法对于链路固有时延变化问题的解决依赖于时间窗口（行⑭~⑲）。以 Linux 内核实现为例，BBR 算法会维持一个约 10 s 的时间窗口，当 10 s 内均未发生 *RTprop* 更新时，BBR 算法判断可能出现了链路切换，因此进入 PROBE_RTT 阶段，排空网络中的堆积数据，并以新的测量时延更新 *RTprop*。

上述机制在较为稳定的地面网络表现良好，但对于存在频繁链路切换的 LEO-SN，BBR 算法对新链路状态的响应延迟最大可达 1 个时间窗口，即 10 s。在未进入 PROBE_RTT 阶段前，发送方都无法适应新的网络状态，并且会长时间持续出现带宽下降问题。本文将这种卫星切换事件后带宽长时间下降的现象称为“长波谷效应”。

2.2 “长波谷效应”验证

在 Linux 系统环境中使用 LeoEM 模拟器模拟星链 LEO-SN，在 150 Mbps 带宽下，进行 TCP 数据传输实验。在 1 400 s 的时间内，实验观察到共发生 14 次“长波谷效应”。以其中 1 次“长波谷效应”为例，如图 1 所示，长波谷维持时间约为 14 s，期间，平均带宽仅达到 55 Mbps，带宽利用率仅为 37%。

在第 270 秒时，BBR 算法进入 PROBE_RTT 阶段，进行了 1 次链路固有时延探测。在第 280 秒，即 10 s 后，其进行了第 2 次链路固有时延探测。之后，在第 281 秒出现了卫星切换事件，链路固有时延显著增大，BBR 没有探测到新的时延，仍然以旧的时延作为网络参数。此时的高时延被 BBR 误判为由网络拥

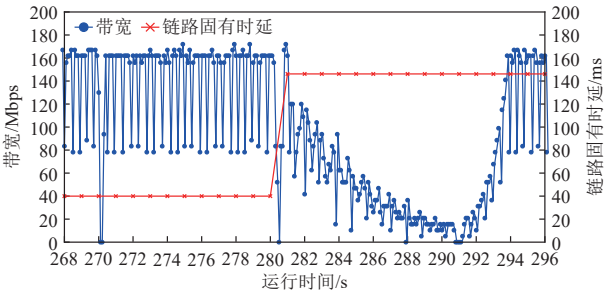


Fig. 1 Bandwidth profile of the long trough effect segments

图 1 “长波谷效应”段的带宽图

塞引起, 进而不断减小发送速率, 试图恢复时延至低时延。因此, 卫星切换事件后出现了明显的“长波谷效应”, 直到第 291 秒重新探测时延后逐渐恢复到最大吞吐。

由此可见, 卫星切换事件会对吞吐量表现带来显著影响。LEO-SN 场景下“长波谷效应”的产生是由短时延链路至长时延链路切换, 而 BBR 算法不能及时更新时延估计引起的。根据本文实验数据统计, 引发“长波谷效应”的 *RTT* 跃迁均落在增幅 54%~79%。在 5 728 s 的 Starlink 低轨卫星网络仿真运行中, 共有 248 次卫星链路切换, 其中有 55 次切换后 *RTT* 跃迁达到 54% 以上, 即卫星链路切换时“长波谷效应”触发概率为 22%, 平均每 104 s(小于 2 min)发生 1 次, 说明该现象具有较高的普遍性。卫星切换事件与“长波谷效应”紧耦合, 且具有一定的周期性, 因此本研究拟设计卫星切换事件的预测模型, 基于卫星切换预测设计一种新的 TCP 拥塞控制机制。

3 SHP-BBR 设计

本文提出了一种基于卫星切换预测的拥塞控制机制 SHP-BBR, 整体设计如图 2, 旨在提升 LEO-SN 中的 TCP 传输带宽利用率。

SHP-BBR 机制通过构建基于门控循环单元(gated recurrent unit, GRU)的卫星切换预测模型, 利用历史时延和切换事件数据, 准确预测即将发生的卫星切换。通过引入注意力(Attention)机制, 模型能够动态突出关键时间步的信息, 从而优化预测效果。在此基础上, 机制结合改进的 BBR 拥塞控制算法, 能够在检测到卫星切换事件后迅速调整拥塞窗口和发送速率, 以适应链路的时延和带宽变化。这样的动态调整策略确保了在高动态环境中数据传输的稳定性和效率, 能够消除“长波谷效应”, 解决由卫星切换引起的带宽下降问题。

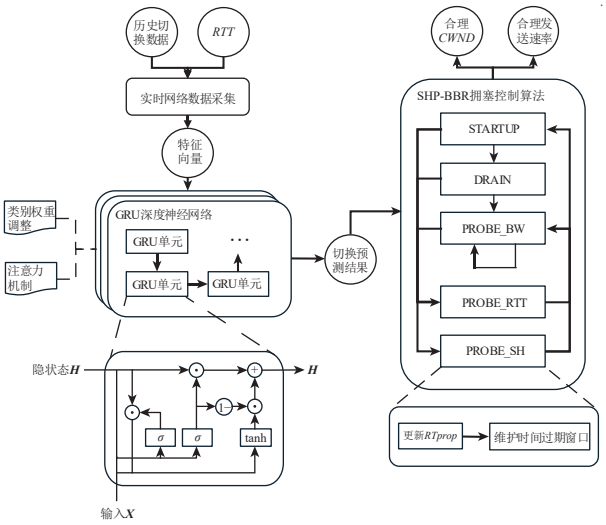


Fig. 2 Overall design of SHP-BBR

图 2 SHP-BBR 整体设计

3.1 卫星切换预测模型

准确预测卫星切换事件是本文所提出 TCP 拥塞控制机制的重要基础。本研究提出了一种基于 GRU 的卫星切换预测模型, 通过综合当前网络状态及其与历史数据的关系, 有效预测未来的切换事件, 从而为基于预测的 TCP 拥塞控制机制提供支持。

首先, 卫星切换预测模型的输入特征由一维数组构成, 包含过去 15 s 内的网络时延数据、卫星切换的历史记录以及当前的实时延迟数据, 共计 31 个元素。时延数据可以由 ACK 获取, 卫星切换的历史记录则可以通过 ICMP 获取路由路径后对比节点变化获得。这些特征的选择充分考虑了卫星网络的时延波动及切换历史信息, 以确保模型可以准确捕捉网络状态的变化。具体来说, 前 15 s 的时延数据用于揭示网络的历史表现, 从而帮助识别潜在的时延波动和性能变化; 而前 15 s 的历史卫星切换数据则提供了切换事件的上下文, 使模型能够学习到先前切换行为对当前网络状态的影响; 最后, 当前的时延数据直接反映了网络的实时状态, 便于模型根据最新的信息进行动态调整。

为了生成准确的输入特征和标签, 本研究采用了滑动窗口技术。在原始时延数据流和卫星切换历史数据中, 滑动窗口技术用于提取特征。通过汇聚过去一定时间范围内的信息, 模型能够优化对当前网络状态的描述。输出标签定义为目标时间步的切换状态, 二元分类的结果为 0(无切换事件)或 1(切换事件)。这种设计使得模型能够有效学习切换行为的产生与影响, 从而提高预测的准确性。

模型结构设计如图 3 所示, 核心部分为 2 层 GRU

网络。第 1 层 GRU 包含 64 个隐藏单元，它不仅捕获输入序列中的时序特征，还生成全时间步的输出。这一过程将输入序列转化为隐藏状态序列，令模型保留之前信息的时间依赖性。随着输入信息的更新，网络得以基于历史上下文进行更好的预测。接下来，第 2 层 GRU 将进一步处理第 1 层输出的隐藏状态，促进更深层次的特征提取，增强模型对时间依赖性信息的感知。

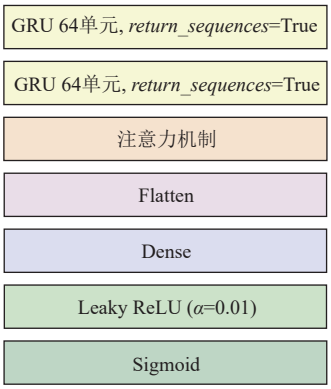


Fig. 3 Structure of satellite handoff prediction model

图 3 卫星切换预测模型结构

为了提高模型在不同时间步上的学习能力，本研究在 GRU 网络中引入了注意力机制。该机制通过对每个时间步的重要性进行动态加权，使模型能够聚焦在预测过程中关键的时间步上，从而优化信息提取。注意力机制层根据 GRU 层的输出权重，为输入的不同时间步分配不同的影响力。该层整合出一个单独的信息向量，突显关键时间步在预测中的重要性，能有效提升模型的预测精度。

在对 GRU 输出进行特征提取后，模型使用 Flatten 层将高维输入展平，转换为一维数据，以便后续的 Dense 层进行处理。在这一步骤中，高维特征被压缩并整理，以为最终的输出做好准备。Dense 层的工作是进一步提取细化后的特征，并将其制备为最终决策的输出。在 Dense 层中，采用了 Leaky ReLU 激活函数，以引入非线性决策边界，并增强模型的学习能力。设置 Leaky ReLU 的 α 值取为 0.01，可以避免神经网络中出现大量死神经元的现象，使得模型在训练过程中保持良好的活性。

模型的最后一步通过 Sigmoid 激活函数处理，以实现二分类输出。此输出映射将预测结果归纳到 2 个状态，分别对应卫星切换事件的发生与否。整个模型构建的过程中，输入层、GRU 层、注意力机制层、Dense 层通过高度的模块化整合，使得模型能够灵活处理多维时序数据，并在复杂的动态环境中做出有效的预测。

在模型的训练过程中，本研究采用了 EarlyStopping 回调机制，以监控验证集的损失情况。设置该机制的主要目的是防止过拟合，并在验证集损失连续 15 个 epoch 内未出现下降时及时停止训练。这一过程确保了最佳模型的训练效果，并回收模型在验证过程中的最佳权重。此外，由于卫星未切换的样本数量远多于发生切换的样本数量，模型面临着严重的类别不平衡问题。考虑到卫星切换事件具有一定的周期性，少数类样本（切换状态为 1）与多数类样本（切换状态为 0）的数值比例为一个较稳定的常数，故而通过类别权重方法提高了对少数类的关注，旨在提升模型在预测切换事件时的准确性和可靠性。

此外，本研究还使用交叉验证技术对模型的表现进行全面评估，在不同划分的训练集和验证集上测试模型，从而确保模型的能力和稳定性。通过对模型进行仔细的调优和细致的验证，本研究构建的卫星切换预测模型能够在复杂的 LEO-SN 环境中实现更高的预测准确度，为后续的 TCP 拥塞控制策略提供坚实的支撑。

总体而言，本研究提出的基于 GRU 的卫星切换预测模型通过深度神经网络架构以及有效的训练策略，为 LEO-SN 的卫星切换事件提供了一种实用高效的预测工具。该模型不仅能够更好地捕捉网络状态的变化，还能为实时调整 TCP 拥塞控制策略提供重要依据，确保其在高动态环境下对网络探测的及时性和准确性。

3.2 基于预测的 BBR 算法优化

本研究基于 BBR 算法进行优化，核心思路是依据 3.1 节提出的卫星切换预测信号，对切换后链路的状态实时监测与分析，动态调整 TCP 的拥塞窗口和数据发送速率，实现高效且可靠的数据传输。BBR 的主要设计思想是通过实时测量网络的瓶颈带宽和最小 *RTT*，以动态调节 TCP 的数据流发送速率。BBR 整个运行过程可划分为多个阶段，包括 STARTUP, DRAIN, PROBE_BW, PROBE_RTT，面向不同传输状态开展拥塞窗口调整。SHP-BBR 在原有 BBR 的基础上新增了 PROBE_SH 阶段，目的是适应频繁出现的卫星链路切换事件，如图 4 所示。

STARTUP 阶段旨在尽快提高数据包的发送速率，以迅速将数据流推向最大可用带宽。DRAIN 阶段则用于快速消除先前数据传输引起的排队累积。PROBE_BW 阶段会定期除去分配给带宽的压力，并通过发送探测数据包对当前的可用带宽进行评估。在以上 3

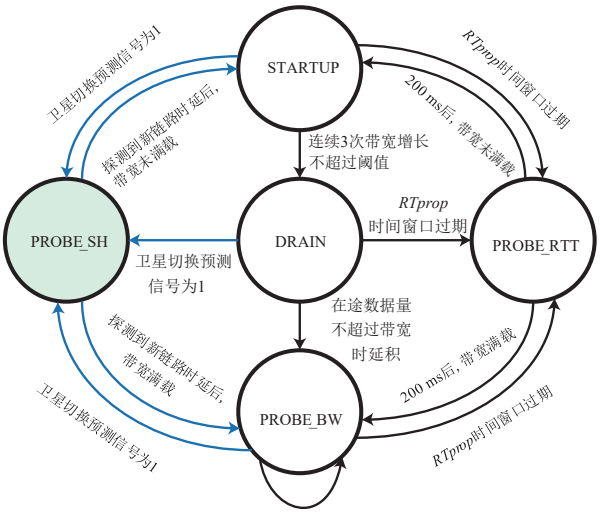


Fig. 4 SHP-BBR state machine

图 4 SHP-BBR 状态机

个阶段, *RTprop* 均遵循最小值更新与时间窗口过期机制, 这意味着其面对卫星切换引起的 *RTT* 剧增时, 无法及时更新链路固有时延估计, 错误判断网络状态。因此, 在以上 3 个阶段中, SHP-BBR 引入了新的跳转机制, 即对卫星切换预测信号的响应。当该信号为 1(即切换)时, 跳转至 PROBE_SH 阶段。而 PROBE_RTT 阶段注重精确测量 *RTT*, 并根据最新的 *RTT* 信息调整最优的拥塞窗口, 减少由于链路波动导致的延迟影响, 故该阶段内不再对卫星切换预测信号做响应。

本文提出的 PROBE_SH 阶段如算法 2 所示。PROBE_SH 阶段在预测到卫星切换事件的情况下, 能对新的网络状态及时、准确估计, 积极调整网络的拥塞控制策略, 保证流量的平稳性和快速响应。

算法 2. PROBE_SH 阶段算法。

输入: *RTT* 测量值 *measuredRTT*;

输出: 拥塞窗口 *CWND*, 发送速率 *pacing_rate*。

① 初始化状态为 STARTUP;
② while 网络正在运行 do
③ if 预测到卫星切换事件 then
④ 进入 PROBE_SH 状态;
/*收集 *RTT* 数据*/
⑤ 维持 *RTprop_update_time* 不变;
⑥ while 未收到新链路 *measuredRTT* do
⑦ 尝试获取新链路 *measuredRTT*;
⑧ end while
/*更新链路估计*/
⑨ *RTprop=measuredRTT*;
/*评估网络状态*/
⑩ 评估网络是否满载;
⑪ if 网络已满 then
⑫ 进入 PROBE_BW 连续探测带宽;
⑬ else
⑭ 进入 STARTUP 最大化传输效率;
⑮ end if
⑯ end if
/*计算 *BDP**/
⑰ *BDP=RTprop*×*BtlBw*;
/*输出 *CWND**/
⑱ *CWND=BDP*;
/*输出 *pacing_rate**/
⑲ *pacing_rate=CWND*/*RTprop*;
⑳ end while

首先, 当卫星切换事件被预测到时, 算法 2 会立刻进入 PROBE_SH 阶段(行③④)。PROBE_SH 阶段的任务是采集新链路的 *RTT* 数据, 确保链路固有时延能够反映实际情况。这一过程会持续到新的链路状态信息被成功采集为止, 通过跳过链路固有时延探测的过期窗口更新机制, 确保系统能够快速进入下一次 PROBE_RTT 阶段以响应时延变化(行⑤~⑧)。在 PROBE_SH 状态下, 系统会强制更新链路的固有时延, 忽略 BBR 算法中对最小时延的限制(行⑨)。这一做法至关重要, 因为卫星切换通常导致链路时延显著增加。如果不及时调整, 传统算法很可能无法快速适应新的网络条件, 从而影响整体性能。

在完成链路更新后, 算法会基于近期的带宽和 *RTT* 信息判断带宽增长是否停滞, 通过 Linux 内核 BBR 标准实现的函数 *bbr_full_bw_reached*() 可以判定网络是否处于满载状态。若网络被认为是满载状态, 系统将返回至 PROBE_BW 阶段, 持续进行带宽探测。如果网络未满载, 则迅速回归 STARTUP 阶段, 以最大化数据传输效率(行⑩~⑮)。

SHP-BBR 会基于链路固有时延与带宽综合估算带宽时延积, 这是控制网络拥塞的基础。链路固有时延的估计是基于一个典型时间窗口内(如 10 s)采集的最小 *RTT*, 这一方法有效消除了瞬时波动对时延计算的影响, 提高了时延信息的代表性。同时, 链路带宽的估计通过在多个 *RTT* 周期内(通常 6~10 个 *RTT*)采集到的最大带宽值进行确定, 这确保了对网络承载能力的全面把握。基于 *BDP* 的实时评估结果, 系统对 *CWND* 和 *pacing_rate* 进行动态调整, 以控制在途数据量(行⑰~⑲)。*CWND* 和 *pacing_rate* 的设置直接关系到 TCP 的数据传输能力, 其调整有助于实现发送速率与网络承载能力的平衡。

本文针对 SHP-BBR 算法的时间与空间复杂度进行分析。首先定义关键参数：设 N 为 TCP 连接期间 BBR 主循环迭代次数（正比于连接时长），S 为卫星切换事件发生次数。时间复杂度取决于 2 种执行路径。

1）正常传输阶段（行②③，⑰~⑲）包含切换预测检测、*BDP* 计算和速率控制操作，单次迭代时间复杂度恒定为 $O(1)$。

2）切换处理阶段（行④~⑮）包含 *RTT* 数据收集（行⑥~⑧）：通过循环等待新链路 *RTT* 样本，耗时取决于网络环境。最坏情况下需等待一个完整 *RTT* 周期（记为 RTT_{new}），时间复杂度为 $O(RTT_{new})$。状态评估与转换（行⑨~⑮）：更新 *RTprop* 估计值并评估网络状态，均为 $O(1)$ 操作。

总时间复杂度由式（6）给出：

$$T_{total}=(N-S)\times O(1)+S\times O(RTT_{new})=O(N+S\times RTT_{new})。\tag{6}$$

实际 LEO-SN 中 S 显著小于 N，RTT_{new} 为有限值且不超过一个完整 *RTT* 周期（毫秒级常数），故 $T_{total}\approx O(N)$，表明算法耗时随连接时长线性增长，不受网络规模影响。

空间复杂度分析表明：算法需维护固定数目状态变量（如 *RTprop*、*BtlBw*、*CWND*、切换信号等，典型值约 80 B）及小型 *RTT* 样本缓存（取决于时间窗口长度，典型值取 10 个 u32 数组加上 2 个 int 类型，约 48 B），总空间复杂度为 $O(1)$。此特性确保算法在终端和无人机节点等资源受限设备中能够高效运行，内存不随网络规模膨胀。

SHP-BBR 通过实时的状态监测，不断评估网络传输的情况，从而优化数据发送速率。通过仅基于链路时延和带宽而制定的动态策略，提升了对发送速率的控制，同时最大限度地提高了数据传输效率。通过灵活调整 *CWND* 与 *pacing_rate* 的结合，用户能够有效降低由于网络拥塞造成的数据包丢失和延迟，从而提升 TCP 连接的整体流畅性。

4 实验评估

4.1 实验环境

实验使用 Linux 环境，基于 LeoEM 模拟器进行 LEO-SN 的模拟。该模拟器不仅可以真实模拟 LEO-SN 的动态特性，还支持全栈仿真，允许其上运行真实的应用程序进行通信模拟。LeoEM 模拟器基于的组件及实验运行的具体环境如表 1 所示。

实验采用 Starlink Shell 1 LEO-SN 的参数进行网络环境模拟。首先基于 StarPerf 组件，根据星座参数计算卫星的动态 3 维位置，构建低轨卫星星座。然后，指定源节点和目的节点位置以及链路类型，并完成路由的预计算。将生成的路径数据输入到仿真进程中，创建 2 个网络命名空间，通过模拟的动态网络使它们相互连接。在 2 个网络命名空间中，使用 iperf 进行网络数据的发送与接收。

Table 1 Experimental Environment

表 1 实验环境

环境配置项	软件版本
操作系统	Ubuntu 20.04.6
内核版本	Linux 5.15.0
星座构建组件	StarPerf v1.0
网络模拟组件	Mininet 2.3.1b4
控制器组件	pox 0.8
发送接收组件	iperf 2.0.13

具体的网络环境参数如表 2 所示。满载带宽设置为 150 Mbps，时延则根据实际路由路径计算得出，即发送方与接收方之间路由路径的物理距离与光速的比值。

Table 2 Network Parameters

表 2 网络参数

参数	配置环境
星座名	Starlink
轨道高度/km	550
轨道周期/s	5 731
倾角/（°）	53
相位偏移	1
轨道数	24
卫星数	66
带宽/Mbps	150

4.2 SHP-BBR 原型实现

SHP-BBR 原型实现的整体架构如图 5 所示，整体实现在发送方一侧，接收方采用标准 TCP 协议栈。

在用户空间内，分别实现了在线网络数据采集模块、卫星切换预测模型。在线网络数据采集模块实时收集 iperf 工具输出的 *RTT* 数据，并依据仿真平台预计算路由计算出历史卫星切换数据，生成符合卫星切换预测模型输入格式的特征向量。卫星切换预测模型实现对是否发生卫星切换事件的预测，生成卫星切换预测信号。在内核空间，分别实现了 netlink 套接字模块、SHP-BBR 拥塞控制模块，二者均

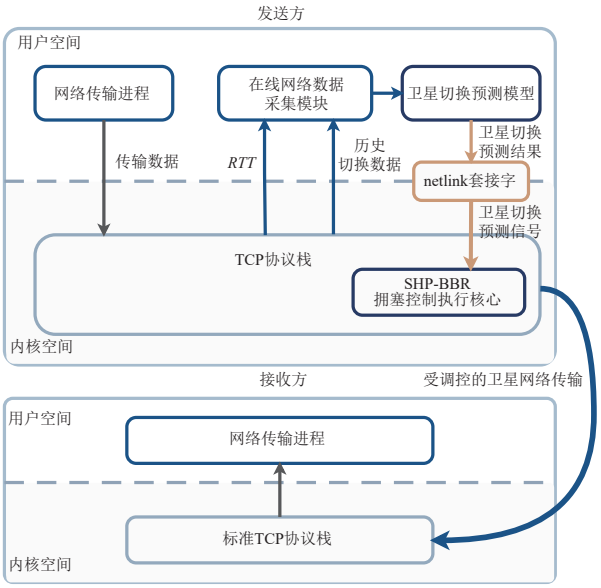


Fig. 5 Prototype implementation of SHP-BBR

图 5 SHP-BBR 原型实现

在仿真开始前加载到内核中。netlink 套接字模块接收用户空间的卫星切换预测信号，通过全局共享变量，传递到内核空间的 SHP-BBR 拥塞控制模块中。基于预测信号，SHP-BBR 模块实现卫星切换时的时延立即更新，调整 *CWND* 以及 *pacing_rate*，实现受调控的 LEO-SN 通信。

4.3 实验结果分析

4.3.1 卫星切换预测模型

本节收集了 LEO-SN 环境中 2 400 s 的 TCP 传输实验数据，将数据集中的 20% 作为测试集，验证卫星切换预测模型的性能，重复实验 5 次，结果如表 3 所示。由实验结果可见，卫星切换预测模型可以准确预测卫星切换事件，可以有效支持 SHP-BBR 机制实现拥塞控制。

Table 3 Satellite Handoff Prediction Results

表 3 卫星切换预测结果

事件	准确率	召回率	*F*1 分数	报告数	真实值
卫星未切换	1.00	1.00	1.00	414	416
卫星切换	0.93	1.00	0.96	27	25

进一步，我们在实验中观察了卫星切换预测模型的漏报、误报对于 SHP-BBR 机制的影响。在时延与带宽表现上，均未有明显的性能下降现象。在漏报的情况下，SHP-BBR 机制对卫星切换感知失效，退化为经典 BBR 算法，不会额外影响传输带宽和时延。在误报的情况下，SHP-BBR 会进入 PROBE_SH 阶段，引入的开销仅为 1 次测量 *RTT* 对 *RTprop* 的更新。然而基于 *BDP* 模型，在未发生卫星链路切换的情况下，SHP-BBR 始终维持路由节点低排队或无排队，因此 *RTprop* 的更新并不会带来网络状态的误判。由此可见，SHP-BBR 对于模型预测失误具有较好的鲁棒性。

本文进一步通过实验分析了模型计算与存储开销。实验平台为 12th Gen Intel® Core™ i7-12700KF，GPU 禁用。模型总参数为 1 680 514，按每个参数为 32 b 浮点数估算，模型大小为 6.41 MB。平均推理时间为 37.57 ms，时间标准差为 4.87 ms。运行前内存使用为 545 MB，运行后内存使用为 571 MB，内存增加 26 MB。综合来看，该模型能够满足卫星网络状态预测与控制决策对实时性的要求，且具有轻量易部署的优势。

4.3.2 BBR“长波谷效应”消除

使用 iperf 进行数据传输，模拟从纽约到伦敦的 1 400 s ISL 链路通信，重复实验 5 次。首先我们对比了 BBR 算法与 SHP-BBR 算法在相同的“长波谷效应”时间段上的表现。因为各段上的表现相似，故取其中 1 次为例进行分析，如图 6 所示。

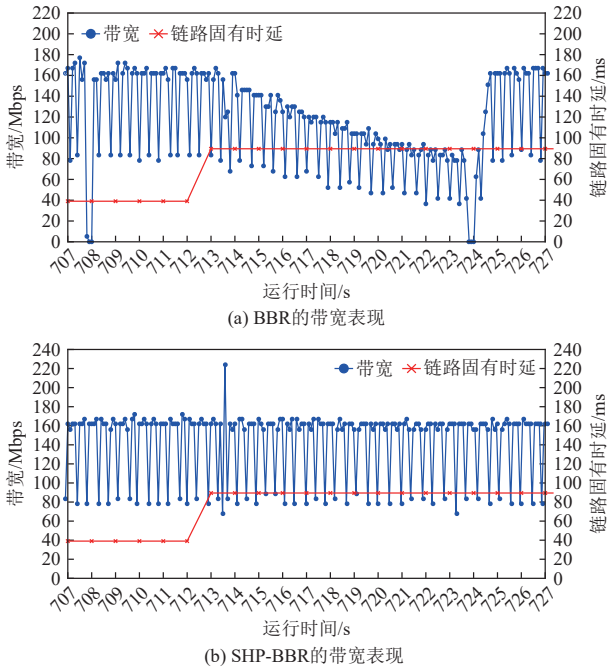


Fig. 6 Bandwidth performance comparison within the same time period

图 6 同一时间段的带宽性能对比

在第 708 秒，BBR 算法进入 PROBE_RTT 阶段，*CWND* 缩小为 4，吞吐量归零，完成了对固有时延估计的一次更新。在第 724 秒，SHP-BBR 进入 PROBE_RTT 阶段，*CWND* 缩小为 4，BBR 算法第 2 次进入 PROBE_RTT 阶段。期间，在第 713 秒，出现卫星切换事件，链路固有时延变化。BBR 未能及时应对，出现“长波

谷效应”。

相比之下，SHP-BBR 在卫星切换预测信号干预下，及时进入 PROBE_SH 阶段，实现了短时间内固有时延估计的快速更新，从而保证了其对网络状态的准确判断，避免了拥塞误判和“长波谷效应”，吞吐量持续保持在 150 Mbps 左右。

实验结果显示，对于 1 400 s LEO-SN 传输内的“长波谷效应”，SHP-BBR 均能有效消除。在发生卫星切换后，SHP-BBR 算法保持近满载的带宽运行，不再出现长达十几秒的带宽波谷。“长波谷效应”的消除对视频通话等实时性应用的服务质量、短时间数据传输的传输性能提升均带来明显收益。

我们统计了 1 400 s 内所有“长波谷效应”时间段内的带宽、时延数据，如图 7(a)(b)所示。在“长波谷效应”时间段内，SHP-BBR 带来了最大 54%、最小 19% 的带宽提升，平均带宽提升 33%。在时延方面，SHP-BBR 带来了 9% 的时延开销。这部分开销的引入是由于 SHP-BBR 更新链路固有时延估计时，为了保证吞吐量收益，未对排队节点进行排空而导致固有时延的测量值偏大，进而影响了准确的固有时延估计。这意味着 SHP-BBR 采用了偏激进的发送策略，尽可能保证带宽的充分利用，以消除“长波谷效应”，但这引入了少量的中间节点排队(即测量 *RTT* 时的中间节点排队数据量)。这种轻微的拥塞引入不会持续太久，在后续的 PROBE_BW 阶段的减周期内，SHP-BBR 算法会逐步通过减小发送速率排空中间节点的数据堆积，同时回归到真实的固有时延估计。

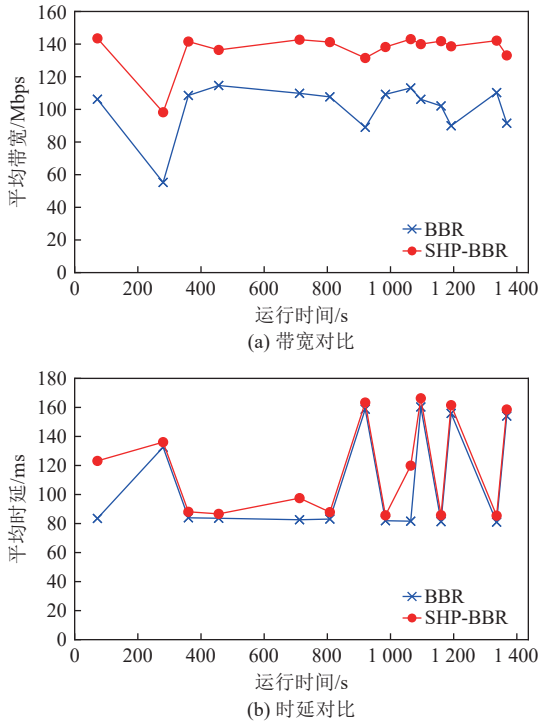


Fig. 7 Bandwidth and delay comparison for all long trough effect segments

图 7 全部“长波谷效应”段的带宽与时延对比

4.3.3 典型 TCP 拥塞控制算法的性能对比

基于 LeoEM 低轨卫星网络模拟实验平台，我们复现了 LeoCC 算法(Github 开源 Live-Network 版本，面向 Linux 内核集成)。由于开源的 LeoCC 算法要求内核允许拥塞控制算法使用较大内存用于存储网络状态和算法状态，无法直接移植到本文实验所用的 Linux 发行版内核，因此，在不改变 LeoCC 算法原有逻辑的前提下，我们通过使用结构体指针增加一级查找逻辑以跳过内核的内存占用空间限制。同时，我们合理简化接收端作为 PoP 节点，用于实现重配置检测功能，以减少对 LeoEM 模拟实验平台的修改要求。基于 Starlink 公司的真实网络传输流量，在上述纽约到伦敦的 1 400 s 星链 ISL 链路环境下，对比测试了本文所提出 SHP-BBR 与 LeoCC 的传输性能。

如图 8(a)所示，LeoCC 在重配置检测命中情况下，与 SHP-BBR 的传输性能表现相当；但是，LeoCC 在部分重配置检测未命中情况下，仍存在类似 BBR 的“长

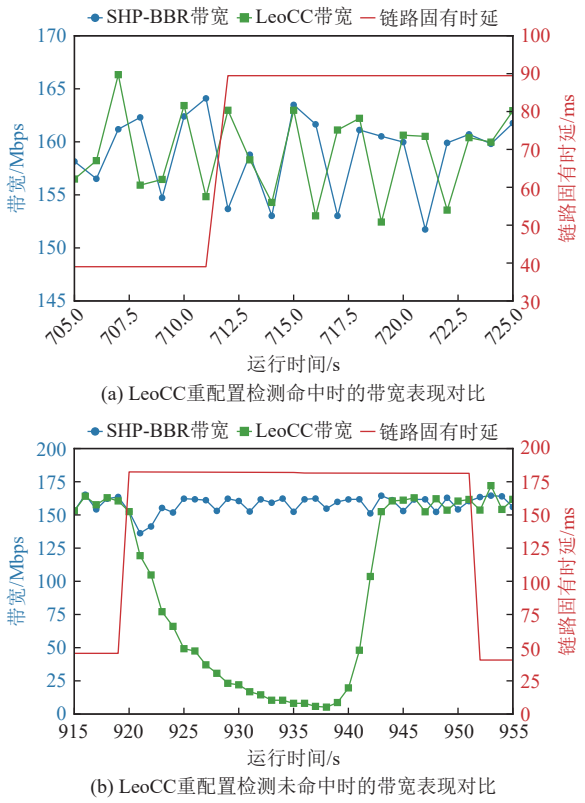


Fig. 8 Bandwidth performance comparison of LeoCC and SHP-BBR

图 8 LeoCC 和 SHP-BBR 的带宽表现对比

波谷效应”的性能下降, 如图 8(b)所示。

图 9 展示了 1 400 s 传输时间段中, LeoCC 和 SHP-BBR 对于卫星链路切换(即网络重配置事件)的检测情况。LeoCC 算法是“反应式”的, 依赖经验设定的固定阈值(实验使用开源代码设定的 45 ms)实现重配置检测, 面向星间、星地不同传输环境, 通常无错报情况但可能有漏报情况。本文所提出的 SHP-BBR 是“预测式”的, 可以通过引入一定的误报代价消除漏报情况, 同时, SHP-BBR 的误报处理不会影响传输性能(如 4.3.1 节所说明)。

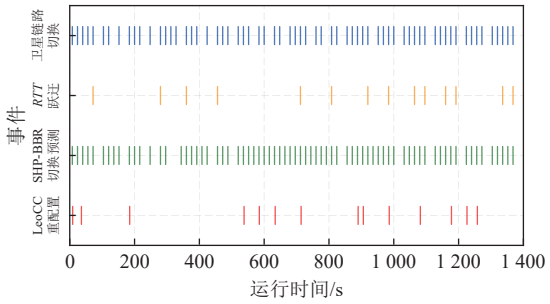


Fig. 9 Comparison of LeoCC and SHP-BBR in detecting satellite link handovers

图 9 LeoCC 与 SHP-BBR 的卫星链路切换检测对比

本文选择了代表性的 TCP 拥塞控制算法和 SHP-BBR 算法进行对比, 分别是基于丢包的 CUBIC、基于时延的 Vegas、面向无线网络优化的 Westwood、基于探测的 BBR。近年来提出其他拥塞控制算法, 例如, Copa 同样以稳定 *RTT* 为假设前提, 时延是其主要拥塞信号, 显然与 LEO-SN 的时延快速变化特性不适应; Deep-BBR 等卫星优化算法建立在 BBR 状态机框架上, 与 BBR 共享相同的时延更新机制缺陷, 不再进行重复的实验对比。

我们模拟了从圣地亚哥到纽约的 1 400 s BP 链路, 重复实验 10 次取平均值, 实验结果如图 10 所示。

在时延表现上, CUBIC, Vegas, Westwood 算法表现优秀, 几乎同样维持在链路的最小时延水平。这是因为三者均始终未能充分利用链路带宽, 不会造成网络拥塞, 进而不会导致时延增大。BBR 和 SHP-BBR 均引入了一定程度的时延开销, 整体表现接近。

在带宽表现方面, CUBIC 的带宽利用率仅约 30%, Vegas 的带宽利用率仅约 13%, 二者几乎不可用。这是由于涉及到卫星的频繁切换, 导致了高频的丢包、固有时延变化, 进而引发 *CWND* 回退。在 LEO-SN 场景下, CUBIC 和 Vegas 算法长期处于上述状态, 因此出现带宽利用率低的表现。Westwood 带宽利用率仅约 52%, 这是因为其稳定 *RTT* 的前提假设被破

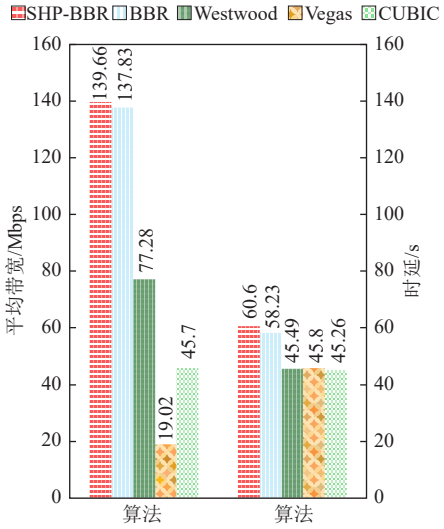


Fig. 10 Bandwidth and delay comparison of various TCP congestion control algorithms

图 10 各 TCP 拥塞控制算法带宽与时延对比

坏, 会把传播时延变化误判为带宽变化。BBR 和 SHP-BBR 算法分别达到了约 138 Mbps 和 140 Mbps, 接近设置的物理链路满载带宽。

综合考虑平均时延和平均带宽, CUBIC, Vegas, Westwood 均存在明显的带宽过低问题, BBR 和 SHP-BBR 协议在 LEO-SN 环境中表现较为平衡, 兼顾了较低的时延和较高的带宽利用率。其中, BBR 在时延表现上略具优势, SHP-BBR 在带宽表现上略具优势。

5 结 论

在本文中, 我们首先分析了接入无人机的 LEO-SN 的特点以及这些特点对传统 TCP 拥塞控制机制造成的影响。通过对现有的 CUBIC, Vegas, BBR 等典型 TCP 拥塞控制机制的原理分析和实验研究, 我们发现其均会遇到明显的性能降级问题。其中, BBR 相较于其他 TCP 拥塞控制机制在 LEO-SN 下略有优势。我们基于 BBR 进一步研究, 通过仿真实验, 发现了 BBR 存在频繁的“长波谷效应”, 即卫星切换后长时间连续的带宽下降。

为了解决这些问题, 我们提出了一种基于卫星切换预测的 TCP 拥塞控制机制, 命名为 SHP-BBR。该机制首先利用历史卫星切换数据和 *RTT* 数据, 通过时间序列预测模型, 准确预测卫星切换事件的发生。预测模型确保了在卫星切换即将发生时, SHP-BBR 拥塞控制机制可以提前感知到网络状态变化。基于切换预测信号, SHP-BBR 能够快速响应, 及时对新的网络状态进行准确估计, 避免因传统拥塞控制算

法的拥塞误判而造成的带宽浪费和数据丢失。实验结果显示，SHP-BBR 可以稳定地消除“长波谷效应”，在 LEO-SN 传输中始终保持近于满带宽的传输性能，带来了平均 33% 的带宽提升，这对实时应用（如视频通话等）和短时间数据传输具有重要的实际意义。与经典 TCP 拥塞控制算法（如 CUBIC, Vegas, Westwood）相比，我们提出的 SHP-BBR 算法在动态变化频繁的 LEO-SN 环境中，能兼顾吞吐量和时延表现。

这项研究为 LEO-SN 的数据传输提供了一种更高效、更稳定的方法，为结合无人机、低轨卫星的“空-天-地”一体化网络的 TCP 拥塞控制机制优化提供了新的思路。在后续的工作中，我们将在更加多样和真实的场景下对现有工作进行性能验证并优化参数设置，进一步提高 TCP 协议在 LEO-SN 中的性能表现。

作者贡献声明：刘永基、李成龙负责方案整体设计、原型实现、实验验证并撰写论文；黄安文、李琼提出指导意见并修改论文；刘洋徐瑞、宋鑫负责部分算法思路和数据分析。

参 考 文 献

[1] Shen Jiawei, Hong Tao, Zhang Gengxin. Access and transmission technology for LEO satellite networks[J]. ZTE Technology Journal, 2024, 30(5): 68−74 (in Chinese)
（申佳伟, 洪涛, 张更新. 低轨卫星网络接入与传输技术[J]. 中兴通讯技术, 2024, 30(5): 68−74）

[2] Zeng Yong, Zhang Rui, Lim T J. Wireless communications with unmanned aerial vehicles: Opportunities and challenges[J]. IEEE Communications Magazine, 2016, 54(5): 36−42

[3] Yan Xiaohu. Research on end-to-end congestion control mechanism of low-orbit satellite network[D]. Beijing: Beijing University of Posts and Telecommunications, 2022 (in Chinese)
（闫晓虎. 低轨卫星网络端到端拥塞控制机制研究[D]. 北京：北京邮电大学, 2022）

[4] Cao Xuyang, Zhang Xinyu. SaTCP: Link-layer informed TCP adaptation for highly dynamic LEO satellite networks[C]//Proc of the 42nd IEEE Int Conf on Computer Communications (INFOCOM 2023). Piscataway, NJ: IEEE, 2023: 1−10

[5] Henderson T, Floyd S, Gurtov A, et al. The NewReno modification to TCP's fast recovery algorithm[R/OL]. 2012[2026-04-19]. https://www.rfc-editor.org/rfc/rfc6582

[6] Ha S, Rhee I, Xu Lisong. CUBIC: A new TCP-friendly high-speed TCP variant[J]. ACM SIGOPS Operating Systems Review, 2008, 42(5): 64−74

[7] Casetti C, Gerla M, Mascolo S, et al. TCP Westwood: End-to-end congestion control for wired/wireless networks[J]. Wireless Networks, 2002, 8(5): 467−479

[8] Brakmo L S, O'Malley S W, Peterson L L. TCP Vegas: New techniques for congestion detection and avoidance[C]//Proc of the 24th Conf on Communications Architectures, Protocols and Applications. New York: ACM, 1994: 24−35

[9] Dong Mo, Tong Meng, Zarchy D, et al. PCC vivace: Online-learning congestion control[C]//Proc of the 15th USENIX Conf on Networked Systems Design and Implementation. Berkeley, CA: USENIX Association, 2018: 343−356

[10] Arun V, Balakrishnan H. Copa: Practical delay-based congestion control for the Internet[C]//Proc of the 3rd Applied Networking Research Workshop. New York: ACM, 2018: 1−7

[11] Cardwell N, Cheng Yuchung, Gunn C S B. BBR: Congestion-based congestion control[J]. Communications of the ACM, 2017, 60(2): 58−66

[12] Wang Zihan, Zhang Jiao, Zhang Yuan, et al. A transport control protocol for low earth orbit satellite networks based on link information estimation[J]. Journal of Computer Research and Development, 2023, 60(8): 1846−1857 (in Chinese)
（王子涵, 张娇, 张远, 等. 基于链路信息估计的低轨卫星网络传输控制协议[J]. 计算机研究与发展, 2023, 60(8): 1846−1857）

[13] Barbosa G, Theeranantachai S, Zhang Beichuan, et al. A comparative evaluation of TCP congestion control schemes over low-earth-orbit (LEO) satellite networks[C]//Proc of the 18th Asian Internet Engineering Conf. New York: ACM, 2023: 105−112

[14] Li Ning, Deng Zhongliang, Zhu Qiaodi, et al. AdaBoost-TCP: A machine learning-based congestion control method for satellite networks[C]//Proc of the 19th Int Conf on Communication Technology. New York: IEEE, 2019: 1126−1129

[15] Qin Jianchang. Research on congestion control algorithm based on network congestion prediction in LEO satellite network[D]. Beijing: Beijing University of Posts and Telecommunications, 2021 (in Chinese)
（覃健昌. LEO 卫星网络中基于网络拥塞预测的 TCP 拥塞控制算法研究[D]. 北京：北京邮电大学, 2021）

[16] Liu Zhongqi. Study on a congestion control mechanism based on congestion level prediction in satellite networks[D]. Xi'an: Xidian University, 2022 (in Chinese)
（刘中祺. 基于拥塞等级预测的卫星网络拥塞控制机制研究[D]. 西安：西安电子科技大学, 2022）

[17] Li Ning, Zhu Qiaodi, Deng Zhongliang. LDM-Satellite: A new scheme for packet loss classification over LEO satellite network[J]. China Communications, 2022, 19(12): 207−215

[18] Wang Zhiyi, Zhao Ya, Han Wei, et al. An improved BBR algorithm with adaptive congestion control for LEO satellite networks[C]//Proc of the 1st Int Conf on Cloud and Network Computing. Piscataway, NJ: IEEE, 2024: 23−29

[19] Durresi A, Sridharan M, Liu Chunlei, et al. Improved explicit

congestion notification for satellite networks[G]//SPIE 4524: Quality of Service over Next-Generation Data Networks. Bellingham, WA: SPIE, 2001: 293−303

[20] Kimura M, Imaizumi M, Nakagawa T. Reliability analysis of window flow control based on packet transmission interval with ECN considering packet loss[J]. International Journal of Reliability, Quality and Safety Engineering, 2021, 28(1): 2150001

[21] Inoue M, Tangdumrongrat P, Prabhavat S, et al. Mark-reverse explicit congestion notification: Concept, analysis and evaluation[J]. IEICE Technical Report, 2012, 112(208): 153−158

[22] Majidi A, Jahanbakhsh N, Gao Xiaofeng, et al. DC-ECN: A machine-learning based dynamic threshold control scheme for ECN marking in DCN[J]. Computer Communications, 2020, 150: 334−345

[23] Lai Zeqi, Li Zonglun, Wu Qian, et al. LeoCC: Making Internet congestion control robust to LEO satellite dynamics[C]//Proc of the 39th ACM SIGCOMM 2025 Conf. New York: ACM, 2025: 129−146



Liu Yongji, born in 1999. PhD candidate. His main research interests include computer networks and satellite communications.

刘永基, 1999 年生。博士研究生。主要研究方向为计算机网络、卫星通信。



Li Chenglong, born in 1995. PhD, assistant professor. His main research interests include network architecture and in-network computing.

李成龙, 1995 年生。博士, 助理研究员。主要研究方向为网络体系结构、在网计算。



Huang Anwen, born in 1983. PhD, associate professor. His main research interests include computer architecture and microprocessor architecture.

黄安文, 1983 年生。博士, 副研究员。主要研究方向为计算机体系结构、微处理器体系结构。



Li Qiong, born in 1967. PhD, professor, PhD supervisor. Her main research interests include high-performance computer system architecture, intelligent computing, and information storage.

李　琼, 1967 年生。博士, 研究员, 博士生导师。主要研究方向为高性能计算机系统结构、智能计算、信息存储。



Liu Yangxurui, born in 1991. PhD, associate professor. His main research interests include computer architecture, large-scale integrated circuit design, and digital signal processing.

刘洋徐瑞, 1991 年生。博士, 副研究员。主要研究方向为计算机体系结构、大规模集成电路设计、数字信号处理。



Song Xin, born in 1995. PhD, assistant professor. His main research interests include satellite communication and channel coding.

宋　鑫, 1995 年生。博士, 助理研究员。主要研究方向为卫星通信、信道编码。