基于加权优先级与数据包到达时间的MP-QUIC调度算法

杜 备[1,2],陈 展[1],余昌武[1],杜伟庆[1],谢肇鹏[2*],陈平平[1]
(1. 福州大学物理与信息工程学院,福建福州 350108;2. 福州大学先进制造学院,福建泉州 362251)

摘 要: 现有多路快速用户数据报协议(User Datagram Protocol,UDP)互联网连接(MultiPath Quick UDP Internet Connections,MP-QUIC)协议的调度算法忽略了流之间的优先级关系,在异构网络中无法有效区分关键流与普通流,导致网页加载的关键流阻塞,严重影响用户体验。因此,本文提出了一种基于加权优先级与数据包到达时间的多路径调度(Priority-Weighted and Packet Arrival Time based Scheduling,PW-PATS)算法,提升了MP-QUIC协议在异构网络环境中的关键业务流和整体业务传输性能。PW-PATS算法通过将快速UDP互联网连接(Quick UDP Internet Connections,QUIC)流的权重量化为优先级因子(Priority Factor,PF),并作为数据包到达时间(Packet Arrival Time,PAT)的计算权重,形成加权包到达时间(Weighted PAT,W-PAT)的核心路径选择准则,将高优先级数据包调度到高信道质量网络路径。基于网页仿真响应的实验结果表明,相较于传统最短时延优先(Lowest Round-Trip Time first,LowRTT)调度算法,所提算法能够显著提升关键流的传输效率。在传统网页访问模式场景下,高优先级超文本标记语言(HyperText Markup Language,HTML)流的完成时间最高可缩短69%,整体页面加载时间缩短了24%;在网页并行加载模式场景下,高优先级层叠样式表(Cascading Style Sheets,CSS)流的完成时间最高可缩短79.8%,整体页面加载时间在不同网络条件下也获得高达48.9%的缩短。

关键词: MP-QUIC;流优先级;异构网络;调度算法;路径调度
基金项目: 国家自然科学基金(No.62171135,No.62571134)
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112(2026)02-0862-13
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20251104

A Priority-Weighted and Packet Arrival Time Based Scheduling Algorithm for MP-QUIC

DU Bei[1,2], CHEN Zhan[1], YU Changwu[1], DU Weiqing[1], XIE Zhaopeng[2*], CHEN Pingping[1]
(1. *School of Physics and Information Engineering*, *Fuzhou University*, *Fuzhou*, *Fujian* 350108, *China*;
2. *School of Advanced Manufacturing*, *Fuzhou University*, *Quanzhou*, *Fujian* 362251, *China*)

Abstract: Current scheduling algorithms for the multipath quick user datagram protocol (UDP) internet connections (MP-QUIC) protocol overlook the priority relationships among streams, making it ineffective in distinguishing critical streams from ordinary ones in heterogeneous networks. This results in the blocking of critical webpage loading streams and significantly impacts user experience. Therefore, this paper proposes a priority-weighted and packet arrival time based scheduling (PW-PATS) algorithm, which enhances the performance of MP-QUIC for critical and overall service transmission in heterogeneous network environments. The PW-PATS algorithm quantifies the weight of quick UDP internet connections (QUIC) streams into a priority factor (PF) and uses it as a weighting factor in the calculation of packet arrival time (PAT), forming the core path selection criterion of weighted PAT (W-PAT). This prioritizes the scheduling of high-priority packets to network paths with higher channel quality. Experimental results based on webpage simulation responses demonstrate that compared to the existing lowest round-trip time first (LowRTT) scheduling algorithm, the proposed algorithm significantly improves the transmission efficiency of critical streams. In the traditional webpage access pattern scenario, it reduces the completion time of high-priority hypertext markup language (HTML) streams by 69%, while improving the overall webpage rendering time by 24%. In webpage parallel loading mode scenarios, it reduces the completion time of high-priority cascading style sheets (CSS) streams by 79.8%, and also achieves an improvement of up to 48.9% in overall webpage rendering time under various network conditions.

Keywords: MP-QUIC; stream priority; heterogeneous network; scheduling algorithm; path scheduling
Foundation Item(s): National Natural Science Foundation of China (No.62171135, No.62571134)

收稿日期:2025-12-31;录用日期:2026-01-26;责任编委:吴大鹏
*通信作者:谢肇鹏

0　引言

近几十年来,万维网(World Wide Web,Web)技术的飞速发展推动了互联网应用的广泛普及,而Web页面加载速度始终是影响用户体验的关键因素之一。研究表明,当页面加载延迟超过阈值时,用户的满意度和留存率将显著下降[1]。然而,随着互联网用户数量的爆炸式增长,传统的传输控制协议(Transmission Control Protocol,TCP)[2]在应对大规模并发请求和高效数据传输方面表现出诸多局限,尤其是队头阻塞(Head of Line Blocking,HoLB)[3-4]问题,严重制约了Web交互的流畅性。为解决TCP的性能瓶颈,Google于2013年发布了快速用户数据报协议(User Datagram Protocol,UDP)互联网连接[5-6](Quick UDP Internet Connections,QUIC)协议。如图1所示,QUIC在传统TCP的基础上,通过基于UDP传输和集成TLS 1.3(Transport Layer Security 1.3)加密,提供了更高的安全性、更强的灵活性和更低的传输延迟,且抗中间盒干扰能力优异。更重要的是,QUIC通过多路复用机制显著缓解了TCP中的HoLB问题,使Web资源加载更加高效。但随着智能移动设备的普及,终端通常同时拥有多个网络接口(如Wi-Fi[7]、5G[8-9]和卫星网络[10]等),用户对多路径并发传输的需求日益增长,标准QUIC的单路径传输模式在此场景下应对能力不足。

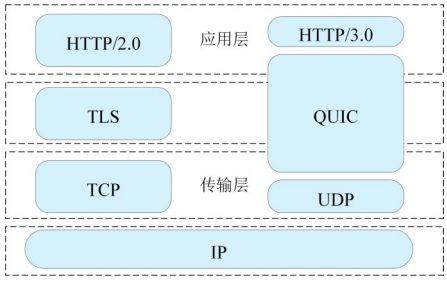


图1　TCP与QUIC协议栈对比

Figure 1　Comparison of TCP and QUIC protocol stacks

在此背景下,De Coninck等人[11]在QUIC协议的基础上进行扩展提出了多路快速UDP互联网连接(MultiPath Quick UDP Internet Connections,MP-QUIC)。MP-QUIC[12]继承了QUIC低延迟、高吞吐的优势,并进一步聚合多条路径的有效带宽,显著提升了数据传输效率,加快了Web页面加载速度。

然而,在网络路径条件差异显著的环境下,MP-QUIC仍然存在诸多挑战。一是流间相互阻塞(inter-stream HoLB)问题。当前MP-QUIC调度算法未对QUIC流间的优先级进行有效划分,导致高优先级流可能被低优先级流阻塞,延长了关键资源的加载时间。二是同一流内数据包阻塞(intra-stream HoLB)问题。MP-QUIC参考了多路TCP[13-14](Multipath TCP,MP-TCP)的调度策略,采用最短时延优先(Lowest Round-Trip Time first,LowRTT)算法选择网络路径,即优先选择往返时延(Round Trip Time,RTT)较低的网络路径进行数据传输。在网络路径条件对称的情况下,该策略能够充分利用各路径的带宽并降低页面加载延迟[15]。但当网络路径差异较大时,LowRTT可能导致带宽利用率不足,出现不合理的路径决策,加剧HoLB问题,进而引发重传和乱序重排[16-17],影响用户体验。

针对上述问题,本文提出了一种基于加权优先级与数据包到达时间的多路径调度(Priority-Weighted and Packet Arrival Time based Scheduling,PW-PATS)算法,以优化MP-QUIC在异构网络环境下的性能。该算法将应用层的流优先级信息与网络层的网络路径状态进行协同,综合考虑并优化,实现异构网络环境中关键流量的低延迟与整体吞吐量的高效平衡。PW-PATS能够在数据包级别动态调度决策,既利用了基于数据包到达时间的调度器(Packet-Arrival-Time based Scheduler,PATS)[18]在路径选择上应对异构性的优势,又弥补了其在流感知方面的不足,最终实现整体性能的优化。本文的主要贡献总结如下:

(1)针对现有调度器的流优先级粗糙静态分类的不足,设计了流优先级量化分类以及动态排序策略,通过对数映射模型将超文本传输协议/2.0(HyperText Transfer Protocol/2.0,HTTP/2.0)等协议中的权重转化为优先级因子(Priority Factor,PF),并采用基于动态排序的比例选择策略自适应地识别高优先级流。在MP-QUIC中实现了细粒度、自适应的优先级感知,从源头为缓解流间阻塞问题提供依据。

(2)建立了融合优先级与网络状态的加权包到达时间(Weighted PAT,W-PAT)调度准则。通过将流PF引入路径质量估计,构建了W-PAT度量,使调度器能够智能地为高优先级数据包选择优质路径,从而有效缓解流内HoLB。

(3)通过大量系统仿真验证了算法在异构网络中的综合性能优势。在多种带宽、时延及混合差异场景下的实验表明,相较于LowRTT算法,所提PW-PATS算法在传统网页访问模式下使超文本标记语言(HyperText Markup Language,HTML)关键资源完成时间最高缩短69%,整体加载时间最高缩短24%,在网页并行加载模式下,使层叠样式表(Cascading Style Sheets,CSS)资源完成时间最高缩短79.8%,整体加载时间最高缩短48.9%,有效解决了关键流加速与整体吞吐量平衡的难题。

1　相关工作

MP-QUIC受MP-TCP的启发,其默认调度策略沿

用了 MP-TCP 的 LowRTT 调度算法[19]。在路径条件相对对称的情况下，LowRTT 能够充分发挥各网络路径的低延迟优势，提升数据传输效率。但在实际网络环境中，由于网络路径间带宽、丢包率、时延抖动等特性存在显著差异，LowRTT 仅依据 RTT 选择网络路径，忽略了带宽、丢包率等关键因素，难以充分利用多路径的潜力，甚至可能导致带宽浪费或不合理的路径决策。因此，为了进一步提升 MP-QUIC 在复杂网络环境下的性能，研究人员提出了多种优化调度算法，以期更有效地解决流间和流内 HoLB 等问题。

Yang 等人[20]提出了一种基于数据包无序传输的策略，确保数据包能够按照正确的顺序到达接收端，降低乱序重排的代价。然而，该方案在某些网络条件下仍然可能引发额外的缓冲和重传开销。Rabitsch 等人[21]在最早完成优先（Earliest Completion First，ECF）[22]算法的基础上提出了流感知最早完成优先（Stream-Aware Earliest Completion First，SA-ECF）算法，通过对流完成时间进行估计来优化数据包调度。该方案主要依赖 RTT 作为路径选择的唯一标准，未能综合考虑带宽、丢包率等因素，在高带宽、高 RTT 路径上表现受限，且在复杂网页负载下无法充分利用同构路径[23]。Shi 等人[24-25]提出的灵活流调度机制（Flexible Stream scheduling mechanism，FStream）和基于优先级的流调度（Priority-based Stream scheduling，PStream）算法在流优先级分配方面引入了权重机制，为高优先级流提供更优的调度策略。这两种方案的优先级分配均基于静态网络环境设计，无法适应突发性的网络质量变化。Guo 等人[26]提出解耦多路径调度器（DE-coupled Multipath Scheduler，DEMS）算法，其通过解耦快、慢子流的数据包发送方向（快子流从前向后、慢子流从后向前），利用单向时延估计实现数据同步到达，但其依赖大缓冲区且扩展性受限，难以适应多路径复杂场景。Zheng 等人[27]提出的 XLINK 进一步细化了视频流的优先级分类，关注视频首帧加载时间，能够实现在流内进行快速重发，并结合体验质量（Quality of Experience，QoE）反馈机制优化视频数据的调度策略，但对于一般的 Web 网页下载流量，其调度逻辑的适用性仍然有限。Liang 等人[28]提出的多流分层调度（Multi-Stream Hierarchical Scheduling，MS-HS）算法通过控制数据分配率，并为每条流赋予两个优先级，以加快高优先级流的传输速度。Xing 等人[29]提出的队头阻塞消除调度器（HoL Blocking Eliminating Scheduler，HBES）调度器采用了一种逐包调度策略，以缓解流内 HoLB 以及接收端缓存过载的问题。

Gao 等人[18]提出的 PATS 通过低复杂度计算来估计数据包的最短到达时间路径，有效降低了任务的整体完成时间。然而，PATS 并未对流的优先级进行任何区分，导致高优先级流可能被低优先级流抢占资源，从而未能有效缓解流间阻塞问题。Lee 等人[30]提出基于深度 Q 网络（Deep Q-Network，DQN）的调度器，将时延和吞吐量作为奖励函数，旨在降低视频块的下载时间。然而，这些方法在实际部署中需考虑计算开销和模型训练的复杂性。Wu 等人[31]提出一种基于在线学习与随机调整的多路径调度算法 Peekaboo，其在动态异构网络中表现出良好的自适应能力。然而，该算法仍存在学习延迟高、对初始流量敏感、缺乏离线学习支持等问题，尤其在高速移动或极端同构路径场景下性能受限。

尽管上述调度算法在不同程度上优化了 MP-QUIC 的路径选择和数据传输效率，但仍无法较好地解决流间和流内 HoLB 等关键问题。算法 LowRTT、SA-ECF 过度依赖 RTT 作为路径决策标准，忽略了带宽、丢包率等关键因素，导致路径选择不合理，无法动态适应网络环境。此外，算法 DQN 虽然优化了数据包调度策略，但计算复杂度较高，难以在实际应用中高效部署。因此，本文设计了综合考虑流优先级与网络路径条件的调度 PW-PATS 算法，以解决 MP-QUIC 在异构网络环境下的传输瓶颈，提高 Web 资源加载的效率。

2 PW-PATS 算法

如上所述，传统 MP-QUIC 调度器在异构网络下面临双重挑战：一方面，缺乏流优先级感知机制，无法保障关键流免受普通流阻塞；另一方面，基于单一指标的路径选择策略，在不对称路径上易引发乱序与等待。为同时应对上述挑战，本文提出 PW-PATS 算法，其核心设计思想是将流调度决策与路径选择决策这两个层级解耦，并进行协同优化。

如图 2 和图 3 所示，PW-PATS 将调度决策解耦为两个核心组件：流优先级量化器（Stream Priority Quantizer，SPQ）和加权路径选择器（Weighted Path Selector，WPS）。SPQ 负责将上层应用为 QUIC 流所显式声明

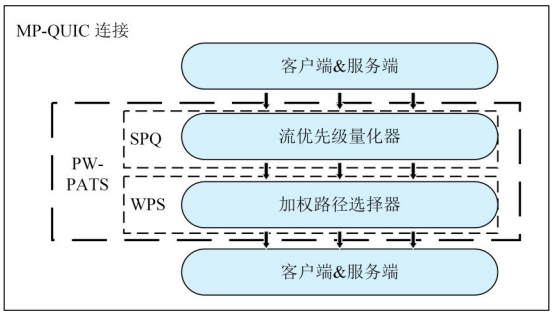


图 2 PW-PATS 结构图

Figure 2 PW-PATS architecture diagram

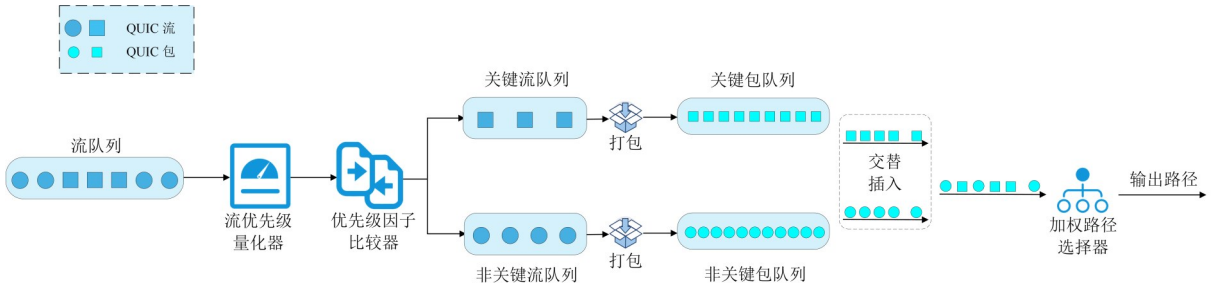


图3　PW-PATS总体流程图

Figure 3　PW-PATS overall flowchart

的优先级权重转化为每个流的PF,并基于动态排序的比例选择策略自适应地识别高、低优先级流。WPS则通过为每条可用网络路径计算W-PAT,并选择具有最小W-PAT值的路径,来执行面向数据包的调度决策。这种分工使得PW-PATS能够将应用层的优先级感知与网络层的路径质量优化集成在一起。本文常用符号含义如表1所示。

表1　符号含义

Table 1　Symbol meaning

符号	含义
Weight^s	QUIC流s声明的权重
PF^s	流s的优先级因子
α	划分高优先级流所占总流的比例
N	当前活跃的路径数量
$S\_\min$	高优先级流最小数量
P	路径集合
p_j	第j条路径
p_{best}	算法选出的最优路径
CW^j	路径p_j的拥塞窗口大小
B^j_{flight}	路径p_j上已发送但未确认的数据量
C^j_{avail}	路径p_j的当前可用发送容量
Q^j_{len}	路径p_j上待发送的数据队列长度
SRTT^j	路径p_j的平滑往返时间
R^j_{wait}	数据包在路径p_j上发送前需等待的RTT轮次
T^j_{queue}	数据包在路径p_j上因排队产生的预计时延
PAT^j	数据包在路径p_j上的原始预计到达时间
W_PAT^j	数据包在路径p_j上的W-PAT
state^j	当前路径p_j的状态(如活跃、空闲等)
min_WPAT	最小的W-PAT值
pf	当前数据包所属流的优先级因子
pkt	当前算法输入的数据包

2.1　SPQ

SPQ是PW-PATS算法实现优先级感知的基础组件。其核心任务是将上层应用中携带的、用于表达传输紧迫性的逻辑优先级权重转化为调度器可理解并可操作的数值化决策依据。与直接将流简单划分为“高”和“低”两个离散优先级类别的粗粒度方法不同,SPQ旨在保留并利用应用层所提供的更丰富、更连续的优先级信息。

2.1.1　PF计算模型

SPQ的输入是上层应用通过类似HTTP/2.0[32]优先级依赖树的机制为每个QUIC流分配的权重。该权重通常为介于1~256的一个整数,数值越高表示分配的相对资源份额越大。然而,若直接将此原始权重应用于路径调度,将引发两个关键问题:其一,数值跨度过大易导致调度决策过度倾斜,高权重流可能过度独占优质路径,破坏基本公平性;其二,线性权重与网络调度中对优先级的非线性感知不匹配。为解决上述问题,并为其后加权路径选择提供精确、稳定的优先级输入,SPQ通过一个映射函数,将原始权重值转化为一个适用于调度的PF,即

$$\text{PF}^s = 1 + \log_2(\text{Weight}^s) \tag{1}$$

其中,Weight^s为应用为QUIC流s声明的权重;PF^s为计算得到的流s的PF。

设计该模型的考量如下:

①非线性映射,应用层权重本身是一个线性值,但在网络调度中对优先级的感知通常是非线性的。使用对数函数可对高权重值进行“平滑压缩”,避免因权重绝对数值差异过大导致调度决策过于激进,破坏基本的公平性。这确保了优先级差异的显著性,同时又保持了系统的稳定性。

②数值范围归一化,通过式(1),PF被规范到一个合理且易于处理的范围内。PF的理论值域为[1,9]。此范围首先服务于高精度流分类,它将应用层分散的权重汇聚到一个分辨率更高的尺度上,使得基于排序和比例的动态分类策略能够灵敏、可靠地识别出真正的高优先级流,从而为从调度源头缓解流间阻塞提供精确、稳定的依据。

为更直观地进行说明,本文考虑一个典型的HTTP/2.0流依赖树结构,如图4所示。在该依赖树

中，每个节点代表一个QUIC流，节点中的数字即为应用层为其分配的权重。

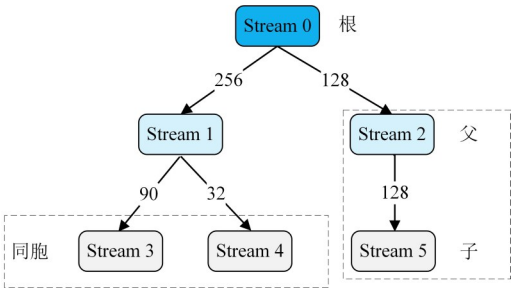


图4 HTTP/2.0流依赖树及权重分配示例

Figure 4 Example of HTTP/2.0 stream dependency tree and weight allocation

依据上述模型，可计算出部分关键流的PF：

Stream1：权重为256，其 $PF^1 \approx 1+\log_2(256)=9$。

Stream2：权重为128，其 $PF^2 \approx 1+\log_2(128)=8$。

Stream3：权重为90，其 $PF^3 \approx 1+\log_2(90)=7.49$。

此计算结果表明，Stream1在调度中将被赋予最高的优先级影响力，而Stream3虽然权重数值远小于Stream1，但其PF值差异被压缩，仍能够保持显著的优先级优势。这验证了计算模型在放大关键流优先级的同时，维持系统整体公平性的设计目标。

2.1.2 高优先级流识别机制

在得出连接中所有QUIC流的PF值后，SPQ需要一种机制来动态识别“高优先级”流。与采用固定阈值的静态方法不同，本文提出一种基于动态排序的比例选择策略。该方法能够更好地适应不同网页或应用所具有的迥异流优先级分布。

其具体流程如下：

①排序与筛选。当连接中创建新的流或现有流的优先级发生变更时，SPQ会收集所有活跃的、有数据待发送的流的PF值，并按降序进行排序。

②动态比例选择。本文定义一个可配置的参数 α，表示被划分为高优先级流的比例，SPQ从排序后的列表顶部开始，选取前 $\alpha \times N$ 个流（N 为当前活跃流总数），将其标记为高优先级流，其余流则标记为普通优先级流。

③最小保障机制。为防止流数量较少时无高优先级流被选中，设置一个最小流数量 $S\_\min$。最终高优先级流的数量为 $\max(S\_\min, \alpha \times N)$。

该动态策略的优势在于其不依赖于绝对的PF阈值。例如，在所有流均为高权重的连接中，它既能够自动提高高优先级流的门槛，又能够在多数流为低权重的连接中确保少数相对重要的流被正确识别。这使算法能够灵活应对多样化的应用场景。此机制完成后，SPQ会维护一个高优先级流列表。当WPS需为数据包查询优先级状态时，SPQ不仅返回其PF值，还将告知其是否属于高优先级类别，为未来更复杂的调度策略提供扩展空间。

2.2 WPS

WPS是PW-PATS算法的执行组件，负责在数据包级别实现基于优先级的智能路径调度，其核心原理是利用SPQ提供的PF，修正传统数据包到达时间模型，构建W-PAT准则，从而为高优先级数据包倾向性地选择优质路径。

2.2.1 原理与计算模型

在MP-QUIC协议框架中，路径的传输能力由其拥塞窗口决定。对于任意路径 p_j，设其拥塞窗口大小为 CW^j，已发送但未被确认的数据量为 B^j_{flight}，且始终满足

$$B^j_{\text{flight}} \leqslant CW^j \tag{2}$$

因此，路径 p_j 的当前可用容量计算式为

$$C^j_{\text{avail}} = CW^j - B^j_{\text{flight}} \tag{3}$$

考虑路径 p_j 的待发送数据队列长度 Q^j_{len}，数据包在发送前的预计排队等待轮数为

$$R^j_{\text{wait}} = \begin{cases} 0, & Q^j_{\text{len}} \leqslant C^j_{\text{avail}} \\ \left\lceil \dfrac{Q^j_{\text{len}} - C^j_{\text{avail}}}{CW^j} \right\rceil, & Q^j_{\text{len}} > C^j_{\text{avail}} \end{cases} \tag{4}$$

基于路径的平滑往返时间 $SRTT^j$，排队导致的时延估计为

$$T^j_{\text{queue}} = R^j_{\text{wait}} \times SRTT^j \tag{5}$$

数据包在路径 p_j 上的原始预计到达时间综合考虑了排队时延和传输时延：

$$PAT^j = T^j_{\text{queue}} + \frac{SRTT^j}{2} \tag{6}$$

WPS的核心创新在于引入PF来修正传统的包到达时间估计。对于属于流 s 的数据包，其加权到达时间W-PAT定义为

$$W\_PAT^j = \frac{PAT^j}{PF^s} \tag{7}$$

该设计确保了高优先级流的数据包具有更小的W_PAT值，使得高优先级流数据包的“感知到达时间”被显著压缩，从而在路径竞争中胜出，被优先调度到优质路径；而低优先级流的调度则近似于标准的PATS算法，保证了基本的公平性。

2.2.2 算法流程

基于上述理论模型，WPS为每个待发送的数据包执行路径选择，其具体步骤如算法1所示。首先，算法初始化最佳路径和最小加权到达时间（行1~2），并计算得出当前数据包所属流的PF（行3）。其次，算法遍历所有可用路径（行4）。对于每条路径，先检查其是否活跃且允许发送，若路径不可用，则跳过该路径

(行5~6)。若路径可用,则根据式(3)计算其可用容量(行8)。再次,算法根据待发送队列长度与可用容量的关系,利用式(4)确定数据包在此路径上的预计等待轮次(行9~13)。利用路径的平滑往返时间(Smooth Round-Trip Time,SRTT)信息,算法依据式(5)和式(6)进一步计算出排队时延和原始的预计到达时间(行14~15)。接着,算法通过式(7)得到该路径的W-PAT(行16)。最后,算法比较所有可用路径的W-PAT,选择值最小的路径作为最佳传输路径(行17~20)。当所有路径评估完毕后,返回所选路径作为数据包的发送路径。

算法1　加权路径选择算法

输入: 数据包pkt,可用路径集合P

输出: 最优路径p_{best}

1. $p_{best}\leftarrow$null
2. min_WPAT$\leftarrow+\infty$
3. pf$\leftarrow 1+\log_2(\text{pkt.stream.Weight})$
4. FOR each $p_j\in P$ DO
5. IF state$^j\neq$ACTIVE or Not SendableTHEN
6. CONTINUE
7. END IF
8. $C^j_{avail}\leftarrow\max(0,\text{CW}^j-B^j_{flight})$
9. IF $Q^j_{len}\leqslant C^j_{avail}$ THEN
10. $R^j_{wait}\leftarrow 0$
11. ELSE
12. $R^j_{wait}\leftarrow\lceil(Q^j_{len}-C^j_{avail})/\text{CW}^j\rceil$
13. END IF
14. $T^j_{queue}\leftarrow R^j_{wait}\times\text{SRTT}^j$
15. PAT$^j\leftarrow T^j_{queue}+\text{SRTT}^j/2$
16. W_PAT$^j\leftarrow$PAT^j/pf
17. IF W_PAT$^j<$min_WPATTHEN
18. $p_{best}\leftarrow p_j$
19. min_WPAT$\leftarrow$W_PAT^j
20. END IF
21. END FOR
22. RETURN p_{best}

3　实验评估

3.1　性能指标

为评估所提PW-PATS算法的性能,本文搭建了基于Mininet[33]的网络仿真测试平台。Mininet能够在一台物理主机上虚拟出完整的网络拓扑,包括交换机、主机及链路,是进行网络协议研究与验证的常用工具。实验所采用的网络拓扑结构如图5所示,该拓扑模拟了一种典型的多路径接入场景:一个配备双网卡的客户端,通过两条异构的网络路径访问一台单路服务器。通过动态调整两条路径的带宽、

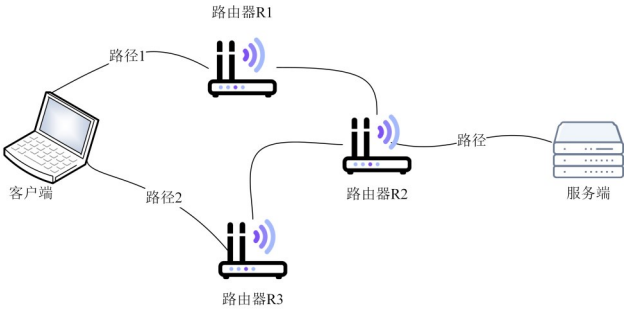


图5　Mininet网络拓扑图

Figure 5　Mininet network topology diagram

延迟等网络状态参数,可构建丰富的异构网络测试场景。

仿真实验中的MP-QUIC通信双方基于XQUIC[34]开源库实现。XQUIC为一个高性能的QUIC协议库,在此基础上扩展实现了PW-PATS调度器以及作为对比的其他调度算法。所有代码均使用C/C++语言编写。仿真平台运行于一台安装Ubuntu 20.04操作系统的笔记本电脑上,其硬件配置为:Intel Core i5-8300H CPU(主频2.30 GHz)及16.0 GB RAM。

3.2　实验设计

为全面评估PW-PATS算法在异构网络下的性能表现,本文设计了两大类共六组实验,从不同维度验证算法在保障关键业务流传输性能方面的有效性。

3.2.1　性能评价指标

实验采用量化指标评估算法性能。

(1)关键资源完成时间。记录HTML文件或CSS文件的完整下载时间,直接反映算法对关键流量的加速效果。

(2)整体任务完成时间。记录所有资源(包括HTML/CSS及附属资源)的完整下载时间,评估算法对整体传输效率的影响。

(3)路径稳定性指数。该指标定义为“总路径使用次数与路径切换次数的比值”,其值越大,表明调度器平均在单条路径上连续发送的数据包越多,决策越倾向于稳定,发生“乒乓切换”效应的可能性越低。

3.2.2　对比算法选择

为进行公平比较,实验选取以下代表性MP-QUIC调度算法作为对比基准。

Lowest Round-Trip Time first(LowRTT)[19]。在发送数据包时,LowRTT遍历所有当前可发送的路径,并估计其RTT,选择其中RTT最小的路径进行发送。

Earliest Completion First(ECF)[22]。ECF首先挑选两条不同路径,其中一条RTT在所有路径中最小但不一定可用,另一条则为可用路径中RTT最小的,然后估计数据包在两条路径上完成传输的时间,并将数据

包调度到完成时间最少的路径上。

Packet-Arrival-Time based Scheduler (PATS)[18]。在发送数据包时,PATS遍历所有当前可发送的路径,并计算不同路径上数据包在接收端的预期到达时间,选择预期到达时间最小的路径进行发送。

Flexible Stream Scheduling Mechanism (FStream)[24]。FStream优先调度时间敏感流,避免流间阻塞,并强制每个流沿单一路径传输,以减少多路径带来的乱序和延迟。同时其通过流优先级比例分配带宽,确保高优先级流的性能。

3.2.3 实验场景配置

网页加载性能的瓶颈通常在于关键渲染路径的优化,其中HTML文件与CSS文件扮演着决定性角色。HTML文件定义了页面的骨架和内容,是浏览器解析和构建文档对象模型(Document Object Model,DOM)树的起点;而CSS文件则控制页面的视觉样式,浏览器必须在下载、解析并构建CSSOM(CSS object model)后,才能与DOM结合形成渲染树,进而进行布局和绘制。因此,CSS文件是一种典型的"阻塞渲染"资源,其下载延迟将直接导致页面加载完成时间延长,严重影响用户体验。基于此,本文将网页加载过程中必须优先保障的HTML流与CSS流定义为"关键业务流"。

为了验证PW-PATS算法在保障此类关键流性能方面的有效性,本文设计了两类实验,分别模拟两种典型的网页加载模式。

第一类实验模拟传统的网页访问模式。首先客户端请求HTML文件(256 KB),客户端接收完成后立即发起一个普通文件(1 MB)的下载请求。在此模式下开展如下测试。

带宽差异测试(场景A)。固定双路径时延均为1 ms,路径1带宽固定为1 Mbit/s,路径2带宽由1 Mbit/s变化至3.5 Mbit/s。

时延差异测试(场景B)。固定双路径带宽均为50 Mbit/s,路径2时延固定为50 ms,路径1时延由50 ms变化至300 ms。

第二类实验模拟网页的并行加载模式。客户端同时发起CSS文件(1 MB)和两个普通文件(各1 MB)的请求,CSS解析完成前页面渲染被阻塞。在此模式下开展如下测试。

带宽差异测试(场景C)。固定双路径时延均为1 ms,路径2带宽固定为1 Mbit/s,路径1带宽由1 Mbit/s变化至25 Mbit/s。

时延差异测试(场景D)。固定双路径带宽为10 Mbit/s,路径2时延固定为1 ms,路径1时延由1 ms变化至500 ms。

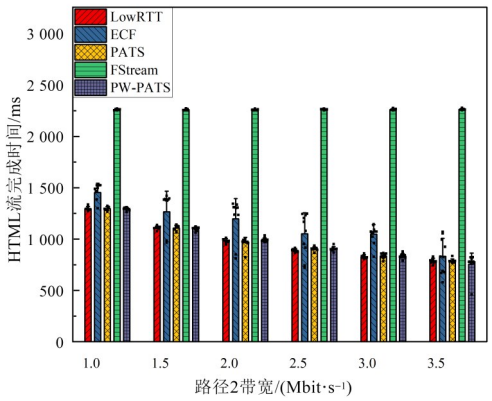
混合异构测试(场景E)。路径1时延固定为200 ms,带宽由1 Mbit/s变化至20 Mbit/s;路径2带宽固定为1 Mbit/s,时延由200 ms变化至1 ms。

可扩展性测试(场景F)。设置三条异构路径,检验算法的可扩展性与稳定性,路径具体参数见下文。

3.3 实验结果分析

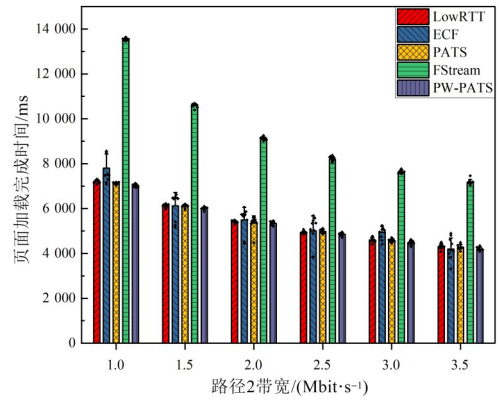
3.3.1 传统网页访问模式下带宽差异对性能的影响

图6(a)和图6(b)分别表示在场景A下各算法的HTML流传输完成时间和页面加载完成时间。



(a) HTML流传输完成时间

(a) HTML stream transmission completion time



(b) 页面加载完成时间

(b) Webpage rendering completion time

图6 传统网页访问模式下带宽异构对完成时间的影响

Figure 6 Impact of bandwidth heterogeneity on completion time in traditional webpage access mode

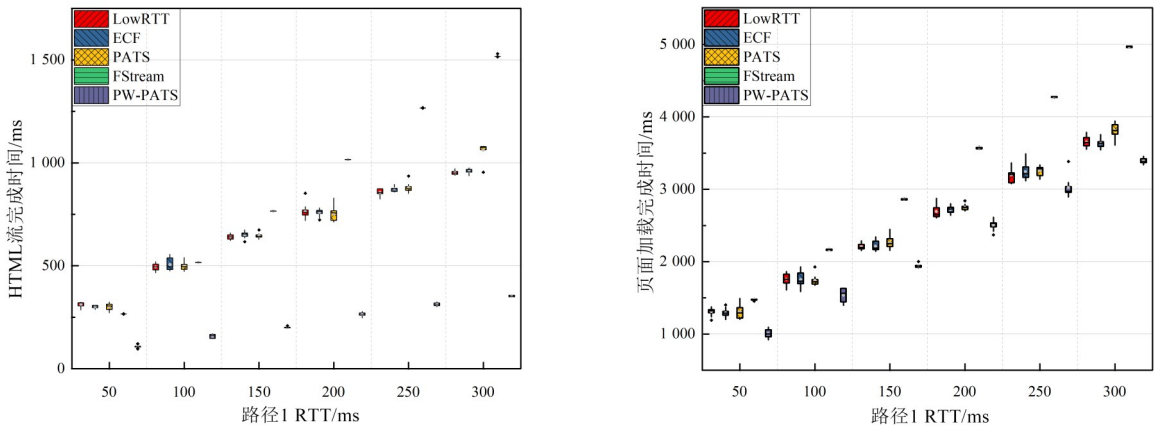
由图6(a)可知,随着路径2带宽的增加,除FStream外,所有算法的HTML流完成时间均呈现下降。FStream性能差的原因主要是其强制每个流沿单一路径传输,在低带宽路径上易出现等待延迟,无法利用多路径缓解带宽不足而导致传输阻塞。ECF算法在低带宽区域调度同样表现不佳。ECF在调度过程中

需进行路径完成时间的预测计算,在数据量较小时所带来的额外开销可能与传输时间本身相当,从而引发调度延迟。此外,ECF在路径选择中倾向于带宽更高的路径,而非延迟更低的路径,在HTML文件传输中可能导致误选路径,从而延长响应时间,且其数据波动较大,不利于稳定传输。相比之下,LowRTT、PATS和PW-PATS算法在HTML流完成时间上表现更为接近,但PW-PATS在多数带宽条件下略优于PATS和LowRTT,显示出其优先级感知机制对关键HTML流的保障效果。

在图6(b)的页面加载场景中,整体趋势与HTML流类似,FStream在全部流完成时间上下降趋势较明显,尤其在带宽提升后改善幅度较大,但其绝对完成时间依然较长。ECF算法仍在低带宽区域表现较差,但在高带宽区域,其表现逐渐优于LowRTT。LowRTT、PATS和PW-PATS在全部流完成时间上的差距相对较小,但在中、高带宽条件下,PW-PATS逐渐展现出更稳定的性能优势,说明其在高带宽场景下能有效分配路径,降低完成时间。

3.3.2　传统网页访问模式下时延差异对性能的影响

图7(a)和图7(b)分别表示在场景B下各算法的HTML流传输完成时间和页面加载完成时间。



(a) HTML流传输完成时间

(a) HTML stream transmission completion time

(b) 页面加载完成时间

(b) Webpage rendering completion time

图7　传统网页访问模式下时延异构对完成时间的影响

Figure 7　Impact of latency heterogeneity on completion time in traditional webpage access mode

由图7(a)可知,随着路径1 RTT的增加,所有算法的完成时间均呈现上升趋势,其中FStream算法在高时延环境下表现明显差于其他算法,尤其是在时延超过200 ms后,其HTML流完成时间急剧上升,并在300 ms时延条件下达到最高。这是由于FStream强制每个流沿单一路径传输,未能充分利用多路径传输的潜力,从而在高时延网络中性能受限。PATS算法在300 ms时,也显露出其调度策略的劣势,这主要源于该算法缺乏对关键流的感知能力。相比之下,PW-PATS性能显著优于其他算法,表现出其出色的流感知机制与路径调度能力。

在图7(b)页面加载场景中,整体趋势与HTML流类似,FStream算法依然在高时延条件下劣势明显,其性能与其他算法的差距巨大。PATS算法由于未引入关键流感知机制,同时其基于数据包到达时间的预估方式增加了计算开销,因此在较高时延下未能取得良好表现。ECF算法在时延差异测试中的表现相对优于其在带宽测试中的表现,虽然仍劣于LowRTT和PW-PATS,但并未如PATS在高时延下出现性能劣化。PW-PATS凭借其在HTML流传输中的显著优势,带动了整体完成时间的最佳表现。由于HTML流作为关键流,其完成时间直接影响后续资源的请求时机,PW-PATS通过优先保障HTML流,使得在页面完成时间上也保持了领先。

3.3.3　网页并行加载模式下带宽差异对性能的影响

图8(a)和图8(b)分别表示在场景C下各算法的CSS流传输完成时间和页面加载完成时间。

由图8(a)可知,PW-PATS的表现明显优于其他算法,其优先级感知机制能够确保高优先级的CSS流优先调度,在低带宽条件下快速完成传输,为后续页面渲染争取了宝贵时间。同样具备优先级感知能力的FStream算法,由于采用每个流沿单一路径的传输方式,表现较差;随着带宽提升,性能虽逐步改善,但仍未超越PW-PATS。其他算法则在低带宽下性能相近,但随着带宽增加,PATS算法逐渐展现优势,在带宽提升至15~20 Mbit/s时,其性能最佳。这主要是因

为PATS基于数据包预估到达时间的调度机制，在该带宽“甜蜜点”能够更精确地协调双路径传输顺序，有效降低了接收端等待时间，从而充分发挥网络容量潜力。然而，当带宽继续增至25 Mbit/s时，PATS的调度优势相对减弱。

在图8(b)中，PW-PATS优先完成了CSS文件的传输，页面渲染过程能够与资源下载同步进行，避免了因CSS解析阻塞而导致的额外延迟。值得注意的是，PATS和FStream在高带宽条件下的表现均优于PW-PATS。这主要由于在高带宽、路径质量良好的环境中，FStream的单一路径绑定策略能有效避免多路径乱序开销，而PATS则凭借其纯粹以最小化包到达时间为目标的调度机制，更高效地利用双路径容量，实现几乎全局最优的包传输序列。然而，这一优势具有明显的场景局限性：FStream在低带宽下性能显著恶化，而PATS则缺乏对关键流的感知能力，在时延较高或流量混杂时调度效果下降(图7)。PW-PATS虽在特定高带宽页面完成时间指标上略逊于二者，但其通过SPQ与WPS的协同设计，在多约束的真实异构网络环境中实现了更稳健的性能优化。LowRTT与ECF整体表现平稳，但在高带宽下仍不及其他算法，说明单纯依赖RTT或完成时间估计，未结合流优先级或包到达时间优化，难以在复杂网络条件下实现最优调度。

3.3.4 网页并行加载模式下时延差异对性能的影响

图9(a)和图9(b)分别表示在场景D下各算法的CSS流传输完成时间和页面加载完成时间。

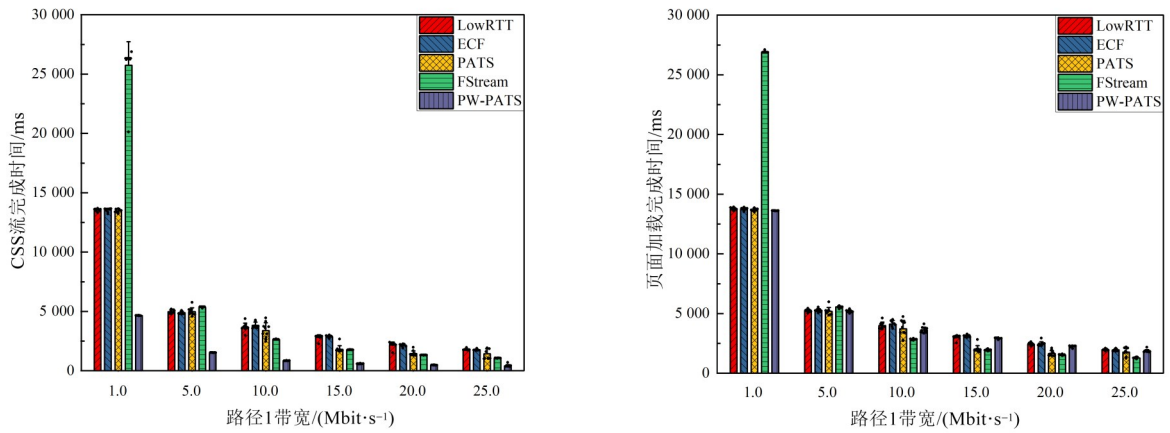
由图9(a)可知，PW-PATS凭借其优先级感知调度机制，始终将高优先级的CSS流调度至当前最优路径，从而在各类时延条件下均保持最低的传输时延。FStream虽然同样具备优先级感知能力，但因其流绑定单一路径的机制，在路径数量少于流数量时会产生资源竞争，尤其在低带宽条件下，这种竞争将引发显著的数据包乱序与处理开销，进而增加传输延迟。随着路径1的RTT由1 ms增加至500 ms，ECF和LowRTT的CSS传输时间显著增长，这是因为依赖固定的路径选择策略，无法动态适应时延变化，导致关键流可能被滞留在高时延路径上。PATS算法表现略优于前两者，因其具备一定的路径质量感知能力，能够依据实时网络状况进行有限度的路径切换，但仍因缺乏明确的优先级区分而无法完全避免性能损失。

如图9(b)所示，PW-PATS由于CSS资源优先完成，页面渲染过程得以尽早启动，实现了资源下载与页面渲染的高效流水线重叠。而其他算法因CSS传输完成较晚，页面渲染需等待关键资源就绪，造成了额外的等待时间。尤其在高时延条件下，ECF和LowRTT的等待时间被进一步放大。PATS虽通过动态路径选择在一定程度上改善了流完成时间，但仍受制于其非优先级调度本质，无法突破由此带来的性能瓶颈。FStream则继续受限于流绑定路径机制，在低带宽条件下性能仍然不佳，难以在高时延场景下实现有效优化。

3.3.5 网页并行加载模式下混合差异对性能的影响

图10(a)和图10(b)分别表示在场景E下各算法的CSS流传输完成时间和页面加载完成时间。

如图10(a)所示，PW-PATS在这些带宽与时延组合下均领先于其他算法，其优先级感知机制能够有效减少CSS文件的传输时间，为页面渲染争取更早的启



(a) CSS流传输完成时间

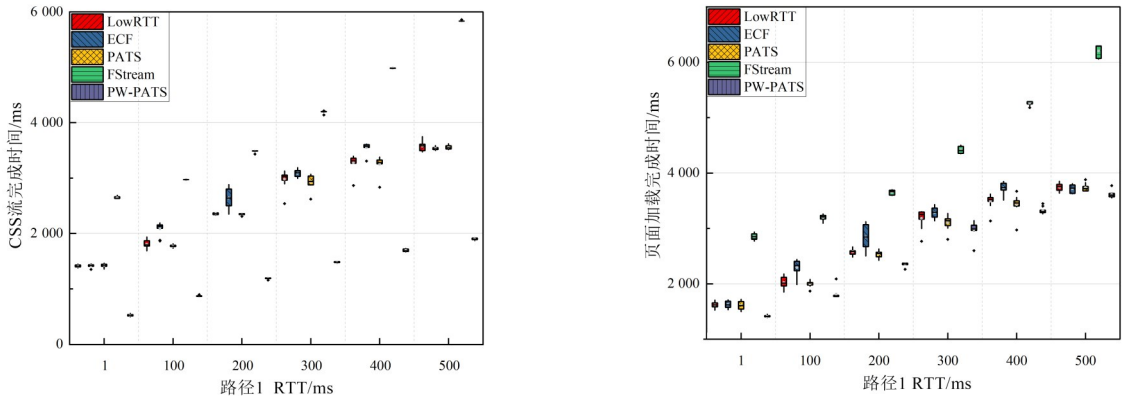
(a) CSS stream transmission completion time

(b) 页面加载完成时间

(b) Webpage rendering completion time

图8 网页并行加载模式下带宽异构对完成时间的影响

Figure 8 Impact of bandwidth heterogeneity on completion time in parallel webpage loading mode

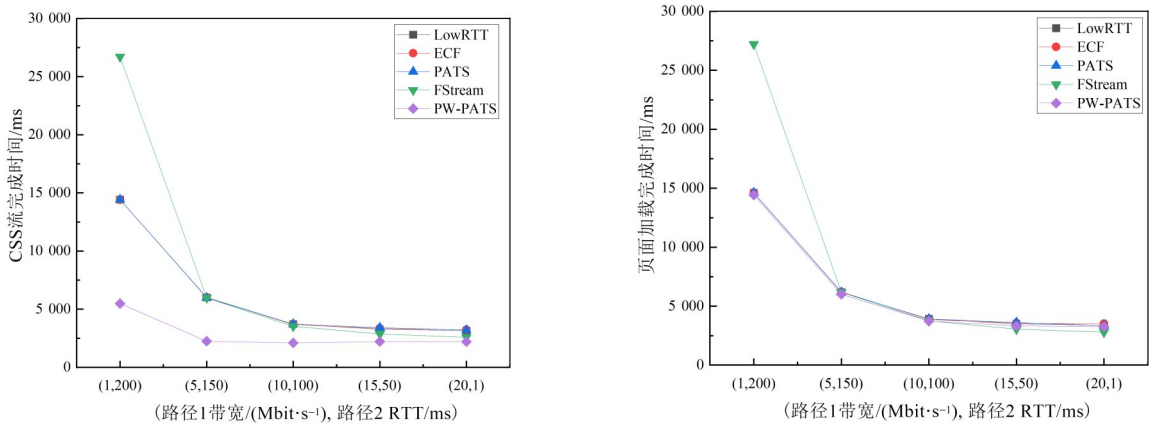


(a) CSS流传输完成时间　(b) 页面加载完成时间

(a) CSS stream transmission completion time　(b) Webpage rendering completion time

图9　网页并行加载模式下时延异构对完成时间的影响

Figure 9　Impact of latency heterogeneity on completion time in parallel webpage loading mode



(a) CSS流传输完成时间　(b) 页面加载完成时间

(a) CSS stream transmission completion time　(b) Webpage rendering completion time

图10　网页并行加载模式下混合异构对完成时间的影响

Figure 10　Impact of mixed heterogeneity on completion time in parallel webpage loading mode

动时机。相对而言,FStream仍受限于其流沿单一路径传输的策略,在低带宽场景中性能下降明显。

如图10(b)所示,PW-PATS虽仍保持整体领先,但其优势相对于CSS传输场景有所减弱。值得注意的是,在高带宽、低时延的网络环境下,FStream表现较好,其页面渲染完成时间甚至优于PW-PATS。主要因为在该条件下,每条路径均具备充足的带宽容量,FStream所采用的路径绑定机制能够有效避免多路径传输可能引发的数据包乱序,同时其基于优先级的带宽分配策略也得以在资源充足的环境中高效运行,从而实现了极低的处理延迟。尽管如此,PW-PATS的整体性能依然展现出更强的适应性与综合优势。其设计初衷并非仅针对高带宽低时延这一特定工况,而是面向真实网络中普遍存在的带宽波动、时延差异及多业务优先级竞争的复杂环境。通过协同优化应用层优先级与网络层路径状态,PW-PATS能够在绝大多数混合异构场景下保持稳定且优异的性能表现,尤其在带宽受限或时延较高的挑战性条件下优势显著。

3.3.6　网页并行加载模式下算法的可扩展性与稳定性验证

如图11所示,本实验设置三条异构路径以验证算法的可扩展性,其参数如下:

路径1:带宽:4 Mbit/s,RTT:50 ms。

路径2:带宽:2 Mbit/s,RTT:30 ms。

路径3:带宽:10 Mbit/s,RTT:100 ms。

每种算法在本场景下均进行10次独立实验以保证结果的统计可靠性。

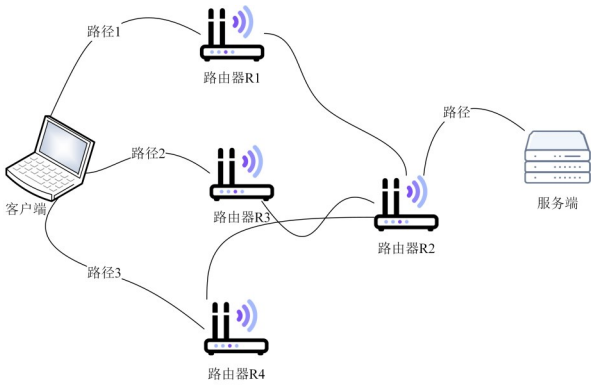


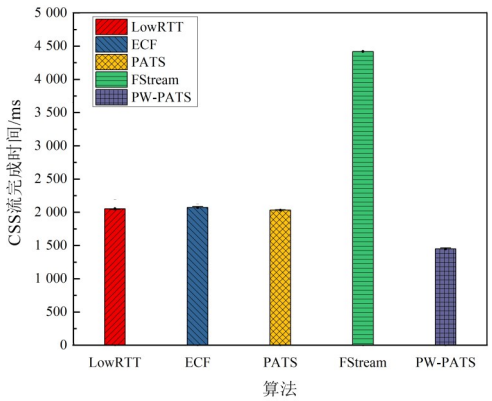
图 11 Mininet 三路径网络拓扑图

Figure 11 Mininet three-path network topology diagram

图 12(a)和图 12(b)分别表示在场景 F 下各算法的 CSS 流传输完成时间和页面加载完成时间。

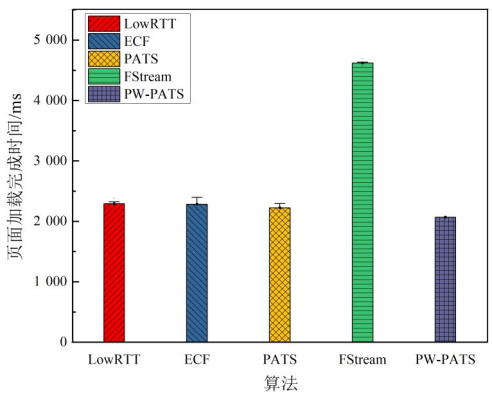
如图 12 所示，PW-PATS 在 CSS 关键流完成时间和整体页面加载时间上均显著优于 LowRTT、ECF、PATS 和 FStream，延续了其在双路径场景下的性能优势。PW-PATS 的 CSS 关键流完成时间比次优的 PATS 算法缩短约 28.7%，整体页面加载时间亦缩短约 7%。这表明 PW-PATS 算法在多路径异构环境下的核心性能优势依然成立。

图 13 表示在场景 F 下各算法的稳定性指数。由于 FStream 算法采用流绑定至单一路径的设计，其稳定性源于固有机制，故不纳入本项对比。



(a) CSS 流传输完成时间

(a) CSS stream transmission completion time



(b) 页面加载完成时间

(b) Webpage rendering completion time

图 12 网页并行加载模式下三路径异构对完成时间的影响

Figure 12 Impact of three-path heterogeneity on completion time in parallel webpage loading mode

如图 13 所示，PW-PATS 稳定性指数与 LowRTT 和 PATS 处于同一水平；而 ECF 算法表现出更高的稳定性指数，这源于其“等待更快路径”的保守策略，该策略虽减少了切换次数，但在动态网络中可能影响负载均衡效率及响应能力。综合来看，PW-PATS 在保持与主流调度算法相当且适中的路径切换频率的同时，实现了最优的传输性能。这证实了 PW-PATS 的加权 W-PAT 准则能够在动态路径选择中保持良好的稳定性，未出现因优先级加权而导致的决策振荡。

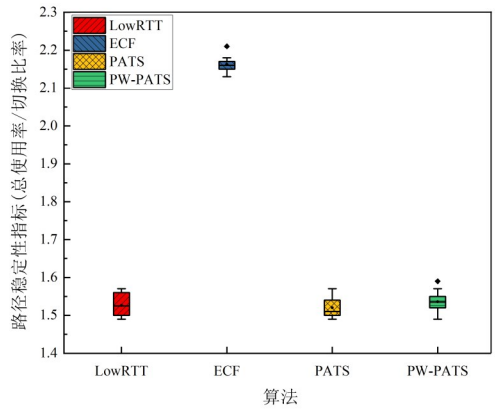


图 13 三路径异构场景下调度稳定性对比

Figure 13 Comparison of scheduling stability in three-path heterogeneity scenarios

4 结束语

本文研究了 MP-QUIC 在异构网络环境下的多路径调度问题，针对现有调度算法中存在的流间和流内 HoLB 等问题，提出了一种基于加权优先级与数据包到达时间的 PW-PATS 算法。通过引入 SPQ 和 WPS 的双层架构，实现了应用层优先级与网络层路径质量的协同优化。仿真实验结果表明，相较于 LowRTT 算法，PW-PATS 在带宽差异、时延差异及混合异构场景下均能显著提升关键流的传输效率，其中在传统网页访问模式下 HTML 关键资源完成时间最高可缩短 69%，在网页并行加载模式下 CSS 资源完成时间最高可缩短 79.8%，同时整体页面加载时间在不同场景

下获得最高24%~48.9%的显著缩短,有效解决了异构网络中关键流加速与整体吞吐量平衡的核心挑战。

尽管PW-PATS在仿真环境中展现出优越性能,但仍存在以下不足:首先,算法对动态网络环境的适应性仍需进一步验证,尤其是在具有高丢包率和频繁波动的真实移动网络中的表现尚待考察;其次,当前优先级映射模型主要基于HTTP/2.0权重体系,对于其他优先级表达机制的兼容性需要扩展;最后,算法在极端网络条件下的公平性保障机制也有待加强。基于当前研究,下一步工作将围绕以下几个方向展开:一是研究动态自适应的优先级调整机制,使算法能够依据网络状态和业务需求自动优化调度策略;二是开展大规模实际网络环境下的部署测试,验证算法在真实应用场景中的性能表现。

参考文献

[1] Schurman E, Brutlag J. Performance related changes and their user impact[C]//Velocity Web Performance and Operations Conference. Sebastopol: O'Reilly Media, Inc., 2009: 1-13.

[2] Tian Y, Xu K, Ansari N. TCP in wireless environments: Problems and solutions[J]. IEEE Communications Magazine, 2005, 43(3): S27-S32.

[3] Khalili R, Gast N, Popovic M, et al. MPTCP is not Pareto-optimal: Performance issues and a possible solution[J]. IEEE/ACM Transactions on Networking, 2013, 21(5): 1651-1665.

[4] Honda M, Nishida Y, Raiciu C, et al. Is it still possible to extend TCP?[C]//Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference. New York: ACM, 2011: 181-194.

[5] RFC 9000 QUIC: A UDP-based multiplexed and secure transport[S/OL]. https://rfceditor.org/rfc/rfc9000.txt.

[6] Cui Y, Li T X, Liu C, et al. Innovating transport with QUIC: Design approaches and research challenges[J]. IEEE Internet Computing, 2017, 21(2): 72-76.

[7] Zreikat A. Performance evaluation of 5G/WiFi-6 coexistence[J]. International Journal of Circuits, Systems and Signal Processing, 2020, 14: 903-913.

[8] 许辰人, 马翔天, 徐昊天, 等. 5G抗干扰技术综述[J]. 电子学报, 2023, 51(3): 765-778.
Xu Chenren, Ma Xiangtian, Xu Haotian, et al. A survey of 5G anti-interference technology[J]. Acta Electronica Sinica, 2023, 51(3): 765-778. (in Chinese)

[9] Andrews J G, Buzzi S, Choi W, et al. What will 5G be [J]. IEEE Journal on Selected Areas in Communications, 2014, 32(6): 1065-1082.

[10] 王朝炜, 杜嘉楠, 王程, 等. 软件定义卫星网络中基于业务调度的混合路由算法[J]. 电子学报, 2024, 52(5): 1506-1515.
Wang Chaowei, Du Jianan, Wang Cheng, et al. A hybrid routing based on traffic scheduling in double-layer software defined satellite networks[J]. Acta Electronica Sinica, 2024, 52(5): 1506-1515. (in Chinese)

[11] De Coninck Q, Bonaventure O. Multipath QUIC: Design and evaluation[C]//Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies. New York: ACM, 2017: 160-166.

[12] Viernickel T, Froemmgen A, Rizk A, et al. Multipath QUIC: A deployable multipath transport protocol[C]//2018 IEEE International Conference on Communications. Piscataway: IEEE, 2018: 8422951.

[13] Peng Q Y, Walid A, Hwang J, et al. Multipath TCP: Analysis, design, and implementation[J]. IEEE/ACM Transactions on Networking, 2016, 24(1): 596-609.

[14] RFC 6182 Architectural guidelines for multipath TCP development[S]. Internet Eng. Task Force, Fremont, CA, USA, 2011.

[15] Paasch C, Detal G, Duchene F, et al. Exploring mobile/WiFi handover with multipath TCP[C]//Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design. New York: ACM, 2012: 31-36.

[16] Sarwar G, Boreli R, Lochin E, et al. Performance evaluation of multipath transport protocol in heterogeneous network environments[C]//2012 International Symposium on Communications and Information Technologies. Piscataway: IEEE, 2012: 985-990.

[17] Paasch C, Ferlin S, Alay O, et al. Experimental evaluation of multipath TCP schedulers[C]//Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop. New York: ACM, 2014: 27-32.

[18] Gao Q, Wang C Y, Yin Y, et al. PATS: A packet-arrival-time based scheduler for MPQUIC[C]//Proceedings of the 3rd International Conference on Networks, Communications and Information Technology. New York: ACM, 2024: 126-131.

[19] Xue K, Chen K, Ni D, et al. Survey of MPTCP-based multipath transmission optimization[J]. Journal of Computer Research and Development, 2016, 53(11): 2512-2529.

[20] Yang F, Wang Q, Amer P D. Out-of-order transmission for in-order arrival scheduling for multipath TCP[C]//Proceedings of the 28th International Conference on Advanced Information Networking and Applications Workshops. Piscataway: IEEE, 2014: 749-752.

[21] Rabitsch A, Hurtig P, Brunstrom A. A stream-aware multipath QUIC scheduler for heterogeneous paths[C]//Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC. New York: ACM, 2018: 29-35.

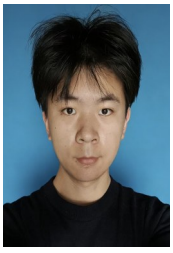
[22] Lim Y S, Nahum E M, Towsley D, et al. ECF: An MPTCP path scheduler to manage heterogeneous paths[C]//Proceedings

of the 13th International Conference on Emerging Networking EXperiments and Technologies. New York: ACM, 2017: 147-159.
[23] Shi H, Cui Y, Wang X, et al. STMS: Improving MPTCP throughput under heterogeneous networks[C]//2018 USENIX Annual Technical Conference (USENIX ATC 18). Boston: USENIX Association, 2018: 719-730.
[24] Shi X, Wang L, Zhang F, et al. FStream: Flexible stream scheduling and prioritizing in multipath-QUIC[C]//Proceedings of the 25th International Conference on Parallel and Distributed Systems. Piscataway: IEEE, 2019: 921-924.
[25] Shi X, Wang L, Zhang F, et al. PStream: Priority-based stream scheduling for heterogeneous paths in multipath-QUIC[C]//Proceedings of the 29th International Conference on Computer Communications and Networks. Piscataway: IEEE, 2020: 9209682.
[26] Guo Y E, Nikravesh A, Mao Z M, et al. Accelerating multipath transport through balanced subflow completion[C]//Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking. New York: ACM, 2017: 141-153.
[27] Zheng Z L, Ma Y F, Liu Y M, et al. XLINK: QoE-driven multi-path QUIC transport in large-scale video services[C]//Proceedings of the 2021 ACM SIGCOMM 2021 Conference. New York: ACM, 2021: 418-432.
[28] Liang X B, Zhao B K, Peng W, et al. Towards effective multipath scheduling with multipath QUIC in heterogeneous paths[C]//Proceedings of the 10th International Conference on Information Systems and Computing Technology. Piscataway: IEEE, 2022: 472-479.
[29] Xing Y T, Xue K P, Zhang Y, et al. A stream-aware MPQUIC scheduler for HTTP traffic in mobile networks[J]. IEEE Transactions on Wireless Communications, 2023, 22(4): 2775-2788.
[30] Lee S, Yoo J. Reinforcement learning based multipath QUIC scheduler for multimedia streaming[J]. Sensors, 2022, 22(17): 6333.
[31] Wu H J, Alay Ö, Brunstrom A, et al. Peekaboo: Learning-based multipath scheduling for dynamic heterogeneous environments[J]. IEEE Journal on Selected Areas in Communications, 2020, 38(10): 2295-2310.
[32] Belshe M, Peon R, Thomson M. Hypertext transfer protocol version 2 (HTTP/2)[EB/OL]. (2015-05-13)[2025-08-04]. https://www.rfc-editor.org/rfc/rfc7540.
[33] Lantz B, Heller B, McKeown N. A network in a laptop: Rapid prototyping for software-defined networks[C]//Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks. New York: ACM, 2010: 1-6.
[34] XQUIC: A high-performance QUIC library[EB/OL]. (2024-04-28)[2025-08-04]. https://github.com/alibaba/xquic.

作者简介

杜 备 男，2000年9月出生于福建省福州市。现为福州大学物理与信息工程学院硕士研究生。主要研究方向为网络传输协议优化、无线通信与MPQUIC调度算法。
E-mail: Dubnium108@icloud.com

杜伟庆 男，1985年10月出生于福建省泉州市。现为福州大学物理与信息工程学院助理研究员。主要研究方向为无线通信、视频编解码与传输。中国电子学会会员编号：E190197703M
E-mail: dwq_qz@fzu.edu.cn



陈 展 男，2002年8月出生于湖南省岳阳市。现为福州大学物理与信息工程学院硕士研究生。主要研究方向为应用层FEC编码、网络传输协议优化与无线通信。
E-mail: chenzhan2002@gmail.com

谢肇鹏 男，1995年7月出生于福建省泉州市。现为福州大学先进制造学院讲师。主要研究方向为信道编码与无线通信等。
E-mail: xzp_fzu@163.com

余昌武 男，2002年4月出生于江西省乐平市。现为福州大学物理与信息工程学院硕士研究生。主要研究方向为前向纠错编码与无线通信。
E-mail: 3538726832@qq.com

陈平平 男，1986年12月出生于福建省泉州市。现为福州大学物理与信息工程学院教授。主要研究方向为信道编码与无线通信。中国电子学会会员编号：E190021215M。
E-mail: t14009@fzu.edu.cn