An Effective Experimental Platform for Multipath Transmission Protocol Algorithms and Performance Analysis

Xuedong Sun[1], Jianxin Wang[1], Neal Xiong[2], Pingping Dong[3]

[1] School of Information Science and Engineering, Central South University, China
[2] Dept. of Business and Computer Science, Southwestern Oklahoma State University, USA
[3] Dept. of Computer Education, Hunan Normal University, China
{xuedongsun, jxwang}@mail.csu.edu.cn, neal.xiong@swosu.edu, ppdong@csu.edu.cn

Abstract

Nowadays, the portable devices with multiple wireless interfaces such as WiFi and 4G/LTE, are becoming more and more popular. However, the legacy transport protocols, such as Transmission Control Protocol (TCP), use only one access interface, ignoring the multipath-transmission opportunity. Although MultiPath TCP (MPTCP) has the potential to improve the transmission efficiency by utilizing multiple wireless interfaces, it has not been a mature technology yet, and suffers a number issues. First, this paper reviews the state-of-the-art of MPTCP. Second, since the simulation result is far from being realistic due to the complicated and dynamic environment in wireless networks, we propose an experimental platform to capture the real performance of MPTCP. Finally, through comprehensive experiments, we compare and analyze the performances of the classical MPTCP congestion control algorithms.

Keywords: Experimental platform, MultiPath TCP, Performance evaluation

1 Introduction

Diverse cloud applications in mobile terminals are extensively used [1], and require stable and high-speed network connections. Most portable devices like smart phones, tablets and laptops have a variety of interfaces (e.g. 3G, 4G, WiFi, IP network). Unfortunately, the legacy transport protocols, such as Transmission Control Protocol (TCP), traverse along one transmission path and uses only one access interface, and is not able to utilize multipath-transmission opportunity in wireless networks [2]. Though there are many redundant transmission paths, these paths are only treated as backup, this is a waste of wireless network resources.

To solve the problem, MultiPath TCP (MPTCP) [3] has been proposed recently by the Internet Engineering Task Force (IETF) working group. MPTCP not only coexists with single-path TCP but also utilizes the multiple interfaces simultaneously. MPTCP is a backward-compatible TCP extension that is capable of using multiple network paths between two end systems for a single TCP connection, with the goal to enhance the performance and the reliability. Due to its various benefits, MPTCP has attracted much attention in research communities. For example, it has been applied in SiRi of iOS7, Samsung Galaxy S6, and Nexus. Besides, Citrix has also implemented MPTCP in the Netscaler load balancing schemes.

While being applied in many applications, the performance of MPTCP is often evaluated by simulations. The simulation environment offers a high degree of network control and repeatable results, which is especially useful in the wireless networks research. However, the simulation environment is typically an abstraction of the reality and therefore contains many simplifications. A wireless networks is a very complicated and dynamic system and cannot be easily captured by the simulation environment. Therefore, the simulation results, many times, do not reflect the measurements in the real platform.

While WiFi and cellular networks are pervasive and extensively used, the two types of the networks exhibit very different characteristics. Cellular networks often show large and varying Round Trip Time (RTT) with low loss rates [4] but WiFi networks have stable RTT with higher loss rates [5]. When MPTCP is deployed in the two networks simultaneously, the network heterogeneity results in several performance issues, including data packet disorder at the receiver, bufferbloat [6], and unbalanced network loads. These issues may degrade the MPTCP performance. To study how a certain MPTCP algorithm performs in the two different networks, we build an experimental platform that incorporates various MPTCP algorithms and allows users to set network parameters, such as packet loss rate, bandwidth, and delay. Therefore, it is a powerful tool to evaluate MPTCP algorithms in different network conditions.

---

[*] Corresponding Author: Jianxin Wang; E-mail: jxwang@mail.csu.edu.cn

DOI: 10.3966/160792642018091905004

Our contributions are as follows. we firstly review the current state-of-the-art of the MPTCP from three aspects, namely, congestion control, flow control and scheduling. Second, we design and implement a experimental platform to evaluate the performance of MPTCP. The experimental platform emulates the wireless network characteristics and implements the classical MPTCP congestion control algorithms, including MultiPath TCP Opportunistic Linked Increases Algorithm (MPTCP-OLIA) [7], Balanced Linked Adaptation Algorithm (Balia) [8], Multipath TCP Veno (MPVeno) [9], and weighted Vegas (wVegas) [10]. Finally, we conduct comprehensive experiments on the proposed experimental platform to evaluate the performance of MPTCP with different congestion control algorithms.

The rest of this paper is organized as follows: Section 2 compares and analyzes the state-of-the-art research of MPTCP. In Section 3, we describe the detail design of the proposed experimental platform, and analyze experimental results. Finally, we conclude this paper with future work in Section 4.

2 State of the Art

In this section, we review the state-of-the-art of MPTCP, and focus on three problems: data package disorder at the receiver side [11], bufferbloat and load balancing problem [12]. The research studies congestion control, scheduling algorithm and flow control.

2.1 Congestion Control

Based on the method of congestion control on the sub-flows of a MPTCP connection, MPTCP congestion control mechanism includes the uncoupled and the coupled algorithms.

In the uncoupled algorithm, each sub-flow performs the congestion control independently, and the mechanism does not consider all network resources. It can be realized by extending TCP Reno, TCP Westwood and other single-path TCP congestion control algorithm. Although uncoupled algorithm can respond flexibly to various changes of the network, it responds so aggressively that it may occupy very much bandwidth in the network path shared with other single-path TCP flows. To solve this problem, a shared congestion detection mechanism has been proposed, namely mTCP [13]. However, the time overhand to identify and suppress sub-flows traversing the shared bottlenecks can not be ignored. Realizing the fact that the shared congestion detection mechanism is slow in responding to the shared bottlenecks, equal weighted TCP (EWTCP) [14] sets a weight for each sub-flow to ensure the fairness. However, the weight given to each sub-flow is the same, without considering the different features of each sub-flow. Thus, the performance of EWTCP becomes worse when each sub-flow has a different RTT.

In the coupled algorithm, the congestion control of each sub-flows is managed in a way that considers the congestion states of all sub-flows. COUPLED [15] makes a multipath flow shift a part of traffic to the least-congested path. However, when the path with the highest loss rate is not the most congested link, COUPLED cannot effectively utilize the network resources. To solve the above problem, the MultiPath TCP Linked Increases Algorithm (MPTCP-LIA) [16] has been proposed. MPTCP-LIA ensures the minimum throughput of MPTCP is no less than that of single-path TCP on the best path, but it is still too aggressive for single-path TCP users. MultiPath TCP Opportunistic Linked Increases Algorithm (MPTCP-OLIA) [7] extends MPTCP-LIA by providing the optimal resource pooling and the batter responsiveness. Most recently Balanced Linked Adaptation Algorithm (Balia) [8] has been proposed to address the MPTCP issues in unbalancing TCP-friendly, responsiveness and window oscillation.

All the above congestion control algorithms are all based on packet losses, but in wireless networks the packet loss is mainly caused by channel attenuation and noises. Considering the characteristics of wireless, several MPTCP congestion control algorithms have been proposed, According to the Westwood fluid model, Multiple Paths TCP Westwood (MPTCPW) [17] promotes the throughput and ensures the fairness by extending TCP Westwood to multipath. Multipath TCP Veno (MPVeno) [9] has been designed based on TCP Veno and adopts the MPTCP-OLIA window increase rule.

The delay-based congestion detection performs better than the loss-based congestion detection in preventing the bufferbloat problem. TCP-Vegas and Fast TCP use RTTs and other delay information to determine the degree of congestion. They avoid the long delay due to the bufferbloat because they throttle the congestion window (cwnd) before RTTs become too large. Based on TCP-Vegas, weighted Vegas (wVegas) [10] uses packet queuing delay to evaluate the available bottleneck bandwidth. Besides, MultiPath Transport Bufferbloat Mitigation (MPT-BM) [18] mitigates the bufferbloat in mobile broadband networks: MPT-BM caps the RTTs by limiting the amount of data in each sub-flow, hence, controlling the bufferbloat. Adaptive Congestion Window MultiPath TCP (ACW-MPTCP) [19] decreases the transmission delay between each sub-flow by adjusting the congestion window of each sub-flow dynamically.

2.2 Flow Control

Flow control and congestion control together govern the transmission rate of a TCP sender and the sending window size is the minimum of the advertised receive window (rwnd) and the congestion window (cwnd).

Therefore, flow control at the receiver can provide an alternative way to prevent bufferbloat. The biggest advantage of receiver-side flow control lies in the faster and easier deployment without the interference of Internet Service Provider (ISP), for preventing bufferbloat.

Through estimating network round-trip time, Dynamic Receive Window Adjustment (DRWA) [20] adjusts rwnd to keep RTT close to the minimum RTT, which is similar to TCP-Vegas. Although the network throughput and delay can be ensured, the performance is not stable enough when it competes with other competitive schemes. Therefore, a Receiver-side TCP Adaptive queue Control (RTAC) [21] is proposed, which controls sending rate by adjusting rwnd in a TCP-friendly manner, according to Active Queue Management (AQM). Based on DRWA, Receive Window Adaptive MPTCP (RWA-MPTCP) is proposed in [22], considering the low packet loss rate in mobile cellular network, observation period for the minimum RTT for better performance and overvaluation of the rwnd size. Through collecting channel information like channel states and signal strength, Available Bandwidth based Receiver Window Dynamic Adjustment (ABRWDA) algorithm [23] retrieves available bandwidth at receiver side directly and uses it and RTT to dynamically calculate rwnd. The adjustment of cwnd will be seen at least after an RTT, while in such period the bandwidth may change. Therefore, a scaling factor λ is set to handle this case. An accelerator to optimize wireless transmission is put forward in [24], which combines random retransmission, flow control and loss packet recovery together. To avoid too much buffer being occupied with one single path, buffer space is allocated to each path equally in [25]. Due to the accuracy of outstanding data for presenting dynamic allocation of system resources, reference [26] replaces occupied buffer space with the amount of outstanding data, based on [25]. And there are relative researches using network queue length prediction [27] and minimum RTT calculation [28] to mitigate bufferbloat.

2.3 Scheduling

To mitigate the impact of packet reordering, some scheduler based different scheduling techniques have been proposed.

2.3.1 Delay Based Algorithms

The default scheduling algorithm is Lowest-RTT-First (LowRTT) in the current MPTCP kernel, which shifts its traffic onto links with smaller RTT. But similar with Round-Robin [29] scheduling algorithm, ACK-Blocking (i.e. data packet is assigned to the sub-flow which receives ACK first) [30] will occur when sub-flows are sent out and the amount of unconfirmed data packet is as large as cwnd. In [31], by developing an analytical model of maximum receiver's buffer blocking time, a Delay-aware Packet Scheduling Algorithm (DPSA) is proposed. In [32], it explains a situation where jitter can occur in an MPTCP data transfer even when underlying network paths are stable. With the idea similar to DAPS, it proposes a new scheduling policy that mitigates jitter by transmitting packets out-of-order on different sub-flows such that they arrive in-order at the MPTCP receiver. Besides in [33], a path selection algorithm is proposed to use Kalman filter to predict transmission delay. In [34], Le and Bui proposed the forward-delay-based packet scheduling (FDPS) algorithm that transmits packets across multiple paths based on delay and bandwidth estimation of various paths in the forward direction (i.e. sender to receiver). In [35], not only the forward delay but also the congestion of sub-flow is being considered to select the forwarding path.

2.3.2 Energy Consumption Based Algorithms

In [36], based on energy models for the different radio interfaces as well as a continuously accumulated communication history of the device user, schedulers were proposed for different applications by solving a Markov decision process offline, which dynamically decides which path, i.e. radio interface, to use for the current active connections. In [37], energy consumption models of TCP and MPTCP on smart phones are developed based on a collected trace, and the eMPTCP algorithm is proposed. To reduce the energy consumption of establishing an LTE sub-flow connection while download small files, eMPTCP delays the establishment of sub-flows over the LTE interface, i.e. eMPTCP does not enable the LTE sub-flow until it receives some bytes over the WiFi interface. The literature [38] considers two mobile applications: real-time applications that have a fixed duration and file transfer applications that have a fixed data size. Then, different optimization is operated according to the energy consumption models of the two applications.

2.3.3 Coding Based Algorithms

Head Of Line (HOL) blocking is a serious problem and aggravates in diverse network conditions. Some researchers combine MPTCP with network coding to mitigating HOL blocking problem. In [32, 39-41], solution that utilizing redundant data to avoid retransmissions in case of delayed or HOL segments was adopted. But due to its fixed coding rate, algorithm performance can not meet the requirements when path quality declines. Therefore, a Fountain code-based Multipath TCP (FMTCP) [42] protocol based on fountain code is proposed, which can transmit encoded symbols from the same or different data blocks over different sub-flows flexibly due to the random nature of fountain code. However, FMTCP does not fully consider the difference of completed time among all sub-flows. Thus, Completion time-aware Flow Scheduling scheme (CaFS) is proposed in [43], which achieves low transmission delay by assigning proper amount of data.

3 The Experiment Platform

The MPTCP verification of MPTCP is often conducted by NS-3 [44-45] and htsim [46] simulation, which can not exactly describe the reaction of MPTCP in the real networks. Bachir Chihani and Denis Collange have implemented MPTCP in the NS-3 to evaluate MPTCP's performance [47]. However, the project has never been merged with any stable ns-3 version and also become obsolete after the TCP stack was rewritten in ns-3.8. In this project only one client could connect to an MPTCP server; i.e. a server could not fork new MPTCP connections. This is a problem particularly when dealing with realistic traffic models. Additionally, the project does not support nodes running TCP and MPTCP at the same time, a feature that is often required while evaluating fairness among competing TCP and MPTCP flows. MPTCP tokens, which uniquely identify the MPTCP connections in a host does not exist. While the MPTCP patch for NS-3 is available in the TCP development communities, the free software suffers the following drawbacks. 1. There are not official uniform standards; 2. There are some problems brought by the different versions of compilation tools; 3. There are errors when one MPTCP connection has too many sub-flows.

To reflect the real performance of MPTCP, and emulate the real wireless network characteristics, we design and implement an experimental platform based on the real network. MPTCP is compiled with a real operating system in the proposed platform, so that the platform can reveal how the algorithms respond when the network changes. The network environment is simulated by a Wide Area Network Emulator (WANem), which is a tool to provide a real experience of a Wide Area Network/Internet. Therefore, our platform allows the MPTCP algorithms to be tested in a realistic WAN environment.

3.1 MPTCP Congestion Control Constraints

MPTCP congestion control should follow three constraints: (1) Improve throughput. MPTCP should at least perform as a single-path TCP running on the best path. (2) Ensure fairness. MPTCP should be TCP friendly, i.e., it should fairly share the bandwidth with the existing single-path TCP on a bottleneck link. 3) Load balancing. MPTCP should move the traffic off its most congested paths as much as possible.

3.2 Platform Design

According to the three constraints of the congestion control algorithm and the emulation settings, a platform based on the real network is established.

3.2.1 Architecture

The platform architecture is shown in Figure 1, which is composed of client clusters, server clusters, and routers. WANem acts as a router, and allows the development team to set up a transparent application gateway that can be used to simulate the WAN characteristics like Network delay, Packet loss, Packet corruption, Disconnections, Packet re-ordering, Jitter, etc. WANem provides a real experience of a Wide Area Network/Internet, and supports the graphical interface to configure the network parameters. The client clusters and the routers are connected to a Gigabit switch. The client gateway is set to be WANem, so all the traffic to the server pass through WANem.

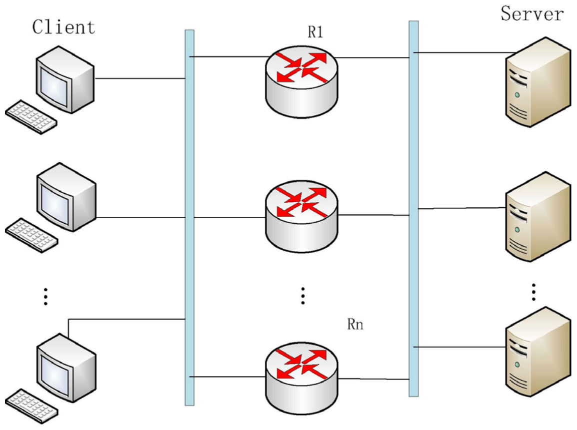
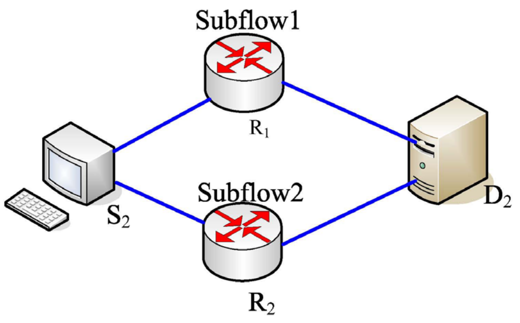


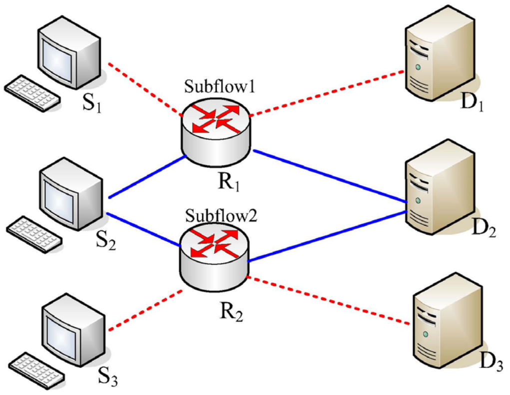
Figure 1. Platform achitecture

We can construct different topologies by using different router rules. According to the three constraints of congestion control algorithm, three experiment scenarios are designed. The experiment scenarios topology is shown in Figure 2, which is composed of three parts according to the three constraints. With different network parameters, different network environments can be emulated and the protocol performance can be evaluated comprehensively. The network parameters are shown in Table 1.

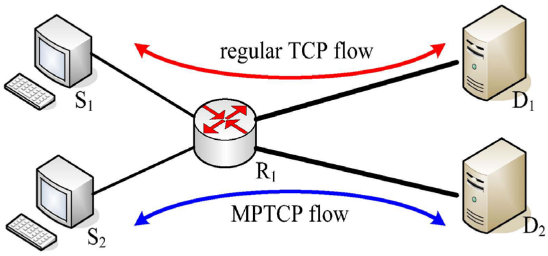
The experiment scenarios is composed of three servers, three clients and two computers with WANem. All the servers and the clients are compiled with Linux Ubuntu 14.04 OS, kernel version 4.1. All the computers are equipped with multiple interfaces. Considering that two interfaces are a popular scenario (e.g., a client with a WiFi and a 4G interfaces), every MPTCP flow has two sub-flows in our platform. The delay, bandwidth, loss rate, and other network parameters are managed by WANem.



(a) Throughput



(b) Load balancing



(c) Fairness

Figure 2. Experiment scenarios topology

Table 1. WANem Network Parameters

network parameters	value
Bandwidth(Mbps)	4,8,16
RTT(ms)	20,40,100,200
packet loss	0%,0.1%,1%,2%,3%,5%

The congestion control algorithms can be implemented as separate Linux modules. After the kernel version 2.6.13, we can implement congestion control algorithms as a pluggable modular on Linux. The *tcp_congestion_ops* structure is initialized first, as show in Figure 3, which defines a number of operating functions of the pluggable congestion control algorithms. And then a system call to the tcp_register_congestion_control is made. In this case a new congestion control algorithm has been registered by the kernel. The next step is to implement the details of the functions included in *tcp_congestion_ops.* We take MPVeno as an example. The congestion control mechanisms are implemented in the *mptcp_mpveno_cong_avoid* () function.

tcp_congestion_ops structure

```
static struct tcp_congestion_ops mptcp_mpveno = {
    .init        = mptcp_mpveno_init,
    .ssthresh    = mptcp_mpveno_ssthresh,
    .cong_avoid        = mptcp_mpveno_cong_avoid,
    .pkts_acked        = mptcp_mpveno_pkts_acked,
    .set_state   = mptcp_mpveno_state,
    .cwnd_event  = mptcp_mpveno_cwnd_event,
    .owner       = THIS_MODULE,
    .name        = "mpveno",
};
```

Figure 3. tcp_congestion_ops structure

3.2.2 Throughput Performance Experiment

In this part, the protocol throughput is measured when the receivers receive the files with different size from the server in different network environment. The topology of this part is shown in Figure 2(a), where multiple MPTCP connections are established and each connection contains two sub-flows. And the routing oolicy has the two sub-flows pass through R_1 and R_2, respectively. Besides, WANem R_1 and R_2 set the network parameters of two sub-flows according to different purposes. To evaluate the algorithm performance when the number of sub-flow changes, the connection and disconnection of two sub-flows can be adjusted in the downloading process. The experiment is carried out with a shell script.

3.2.3 Load Balancing Performance Experiment

In this part, the load balancing performance of each sub-flow in different network environments is tested. As is shown in Figure 2(b), thesurvival time and the number of two single-path TCP links can be managed to construct different network environments. Therefore, the load balancing performance can be evaluated.

Similarly, we set bandwidth, delay and packet loss rate on WANem R_1 and R_2. By using scripts, server S_1 and S_2 can set the number, starting time and finishing time of background flow. When establishing MPTCP connection, it will check if both of the server and client support MPTCP. If so, the MPTCP protocol will be adopted. If not, the single-path TCP will be adopted. Thus, the type of a connection can be controlled through regulating the MPTCP applicability of servers, and the topology can be simplified, i.e. replacing S_1, S_2, and S_3 with S_2 in Figure 2(b). In Figure 2(b), there are three servers D_1, D_2, and D_3, among which D_2 is complied with MPTCP kernel and D_1, D_3 are not. There are two MPTCP sub-flow established from S_2 to D_2, and one passes R_1 and another passes R_2. And the links between S2 and D_1, S_2 and D_3 are single-path TCP. Therefore, connection and disconnection of every link can be controlled in the same client S_2. All operations can be done by using scripts with certain network parameters, which are obtained from terminal.

3.2.4 Fairness Performance Experiment

In this part, whether MPTCP algorithm can fairly shared the bandwidth with the existing single-path TCP on a bottleneck link is being evaluated. As is shown in Figure 2(c), a dumbbell-shaped structure is established. The bottleneck link is being shared by two sub-flow of MPTCP and single-path TCP. Different experimental scenes are set up by altering network parameters and environment parameters, including bandwidth, delay, packet loss rate and amount of links, existing time of links, size of object file. All parameters can be altered by using scripts.

3.3 Evaluation of MPTCP Congestion Control Algorithms

Congestion Control Mechanism is an important factor that affects the performance of the protocol. Thus we implement four classical congestion control algorithms: OLIA, Balia, Mpveno, wVegas in the proposed platform.

In this part, we validate the four existing MPTCP congestion control algorithms in the proposed experimental platform. First, we compare throughput enhancement of the four algorithms. Second, we validate the load balance of four algorithms in a heterogeneous environment in which two sub-flows with different network parameters. Finally, we compare the TCP-friendly of four algorithms when sharing with single-path TCP flows on the same bottleneck link. The platform provides three experimental scripts to set the appropriate parameters of network for different properties in the three experimental scenes, combined with previous experiments designed for MPTCP. A wide range of network environments are considered: the round-trip time varies from 20 ms to 400 ms, packet loss rate is in the range from 0.1% to 5% and the bottleneck bandwidth is also changed from 2 Mbps to 20 Mbps.

3.3.1 Throughput

The topology is shown in Figure 2(a), where ten MPTCP flows are established at the same time, and each MPTCP flow consists of two sub-flows. Two sub-flows have the same settings. GNU Wget is used for generating MPTCP data traffic by downloading documents via HTTP. The document has different sizes from 1MB to 8MB. All the experimental data were collected using tcpdump at the clients, and then analyzed using wireshark. The throughput is calculated as the ratio of file size to download time. The results are shown from Figure 4 to Figure 7. Figure 4 indicates that the algorithms have similar behavior when files sizes are different. Considering that the algorithms are mainly used in the Additive-Increase/Multiplicative-Decrease (AIMD) part of the congestion avoidance phase, we mainly concentrate on steady-state throughput. Thus, we take the 8 MB file as an example in the following experiments.

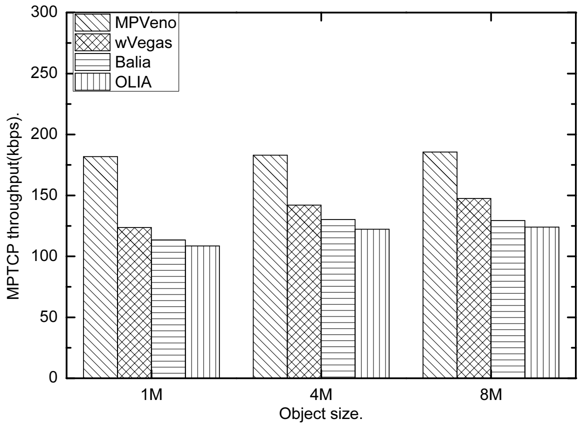


Figure 4. Throughput with varying file sizes. Parameters: $T_1 = T_2 = 100$ ms, $c_1 = c_2 = 8$ Mbps and $p_1 = p_2 = 2\%$

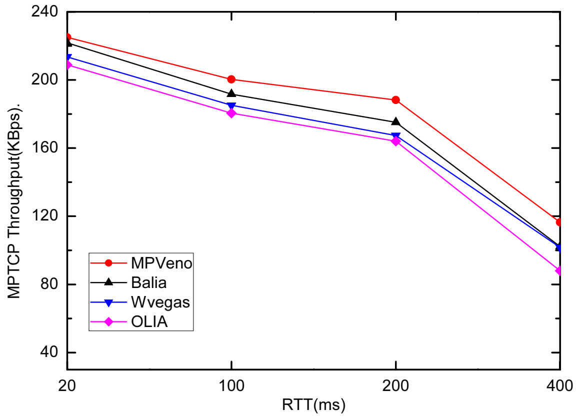


Figure 5. Obtained throughput with varying RTTs. Parameters: $c_1 = c_2 = 8$ Mb/s and $p_1 = p_2 = 0.1\%$

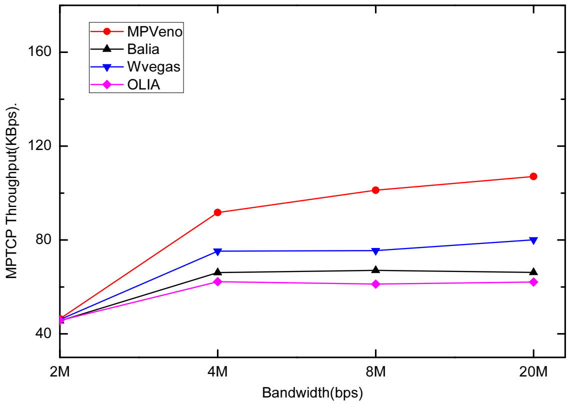


Figure 6. Throughput with varying bandwidth. Parameters: $T_1 = T_2 = 100$ ms and $p_1 = p_2 = 2\%$

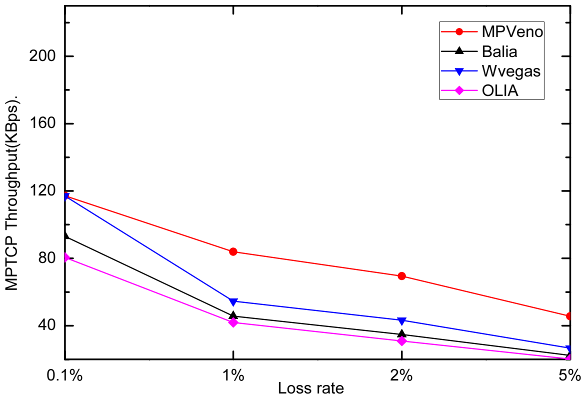


Figure 7. Obtained throughput with varying loss rate. Parameters: $T_1 = T_2 = 400$ ms and $c_1 = c_2 = 20$ Mbps

The experimental results show that MPVeno performs best, followed by wVegas, Balia and OLIA. The reason is that distinguishing causes of the packet loss conduces to MPVeno to avoid decreasing congestion window when the packet loss is caused by random link error [13]. wVegas is a delay-based congestion control, thus it can also survive the random loss rates. Figure 4 indicates that the algorithms have similar behavior when files sizes are different. Besides, when varying RTT, bandwidth and loss rate, we also observe the similar result that MPVeno outperforms other three algorithms, as shown from Figure 5 to Figure 7.

As the number of sub-flows may change during the lifetime of the connection for possible reasons, we construct the environment in which one sub-flow disappears at 60 s. Specifically, after 60s, we close the network interface card using the system command *ifconfig*. For the purpose that MPTCP connection can last long enough (about 300 s), the loss rate, RTT and bandwidth are set as 1%, 200 ms and 2 Mbps, respectively. The throughput is shown in Figure 8. The four algorithms have a similar behavior as the scenario of two sub-flows existing all the time, apart from that wVegas has higher throughput than MPVeno.

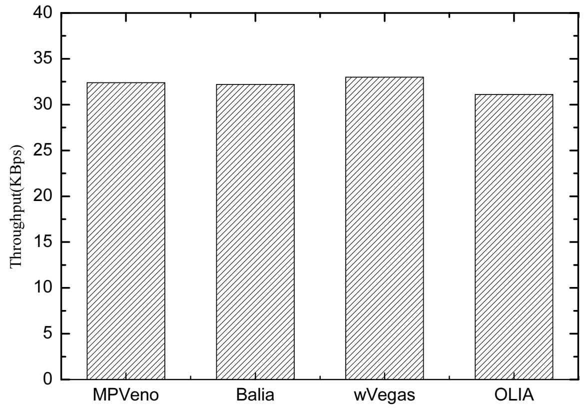


Figure 8. The average throughput (KBps) of the MPTCP flows when one sub-flow disappears at 60 s

3.3.2 Load Balance

The second experimental scene provided by the tested platform adopt the topology shown in Figure 2(b), in order to evaluate the effectiveness of the four algorithms regarding load balancing. In the experiment, two sub-flows have different network configurations to investigate the capability of offloading traffic from more congested path of each algorithm, which have the larger RTT or loss rate. The network configurations are shown in Table 2. The background traffic of Mpveno, wVegas, Balia, OLIA are TCP Veno, Vegas, and Reno, respectively, and the number of flows are varied to generate different traffic loads on each sub-flow.

Table 2. Load Balancing Network Configuration

		Bandwidth (Mbps)	RTT (ms)	packet loss	backflow number
Scenario1	link1	4	100	2%	0
	link2	4	100	1%	11
Scenario2	link1	6	160	0.1%	3
	link2	10	40	0.1%	8
Scenario3	link1	6	160	1%	3
	link2	10	40	1%	8

The two sub-flows have the same bandwidth and RTT in the first experiment. While the packet loss rate is different where Link$_1$ is 2% and and Link$_2$ is 1%. Besides, 11 TCP background flows run on Link$_2$ to guarantee Link$_2$ more congested. The results are depicted in Figure 9. As shown in the figure, the throughput of the sub-flow2 with wVegas algorithm on the more congested Link$_2$ can achieve nearly the same as that of single-path TCP, but throughput of sub-flow1 on the less congested Link$_1$ is not the best. While sub-flow1 of the MPVeno obtains the highest throughput among the four MPTCP algorithms. Meanwhile, the throughput of single-path TCP competed with MPVeno sub-flow1 is no less than that of other algorithms. These results validate that MPVeno can keep balance between sub-flows and achieve a higher network utilization.

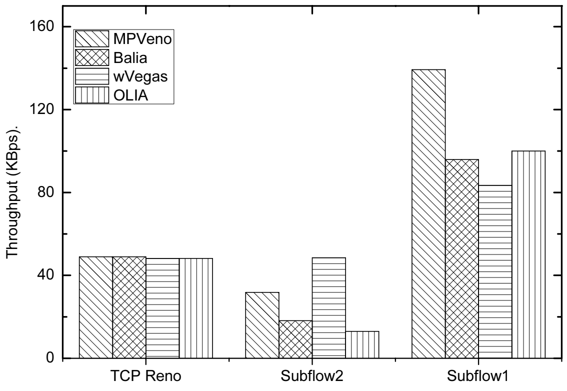


Figure 9. The throughput (KBps) of regular TCP and each sub-flow. Parameters: $T_1 = T_2 = 100$ ms and $c_1 = c_2 = 4$ Mbps

In addition, we also do the experiments scenario2 that the loss rate is the same, and the are RTT and bandwidth are different, as shown in Table 2. To make sure Link$_1$ has a heavy load and Link$_2$ has a light load, there are eight TCP background flows on Link$_1$, and three flows on Link$_2$. We construct two scenarios which have different loss rates. One is 0.1%, and the other is 1%. The results are shown in Figure 10.

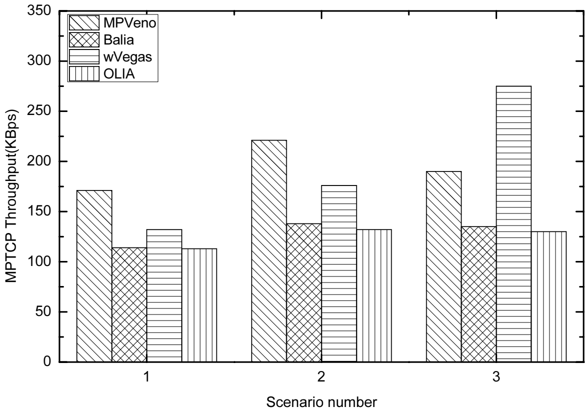


Figure 10. Throughput of MPTCP flows with the adverse network configurations in table 2

3.3.3 Fairness

The MPTCP congestion constraints point out that MPTCP should be TCP-friendly. In this subsection, we investigate that whether the four algorithms can keep fairness when coexisting with single-path TCP flows.

TCP fairness requires that a new protocol receive no larger share of the network than a comparable TCP flow. This is important as TCP is the dominant transport protocol on the Internet, and if new protocols acquire unfair capacity they tend to cause problems such as congestion collapse. Fairness measures is used in network engineering to determine whether users or applications are receiving a fair share of system resources. Raj Jain's equation,

$$J = (x_1, x_2, \cdots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i^2} \quad \textbf{(1)}$$

rates the fairness of a set of values where there are n users and x_i is the throughput for the i th connection. The result ranges from $1/n$ (worst case) to 1 (best case), and it is maximum when all users receive the same allocation. This index is k/n when k users equally share the resource, and the other $n-k$ users receive zero allocation.

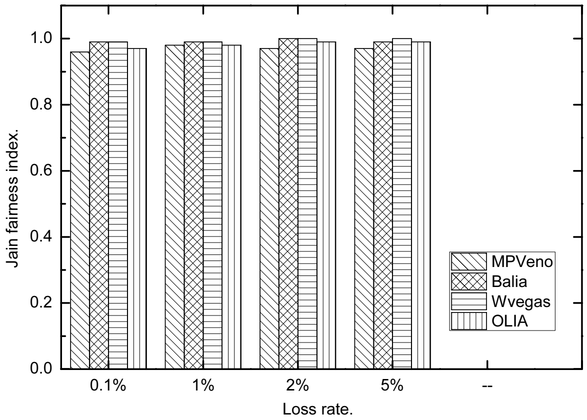


Figure 11. Jain Fairness Index performance. Parameters: T = 400 ms and c= 20 Mbps

The topology for the fairness experiments is shown in Figure 2(c). There are ten MPTCP flows and ten regular TCP flows on the same bottleneck link. The Jain Fairness Index is shown in Figure 11. From the experimental results, we can see that almost all the four algorithms can obtain a high Jain Fairness Index, from which, wVegas performs best and MPVeno performs worst.

4 Conclusion

According to special features of wireless network, MPTCP is proposed to improve the transmission efficiency. We illustrate the related research of MPTCP from three perspectives: congestion control, flow control and scheduling algorithm. To obtain the test results of MPTCP in real-world measurements, we propose the experiment platform, which evaluate the throughput, load balancing and fairness performance of MPTCP congestion control algorithms including MPVeno, wVegas, Balia, and OLIA. The experiment results show that comparing with other algorithms, MPVeno has the larger throughput, the better load balancing, but the fairness needs to be improved. Meanwhile, wVegas and Balia ensure the fairness and the throughput, but the load balancing performance can be optimized. Besides, OLIA has lower performance in load balancing and throughput but better fairness.

The proposed platform has several future work to do as follows.

1. Design more experiment scenes

A series of experiments are designed in this paper, but experiment scenes are relatively simple. More complicated experiment scenes can be designed and implemented based on the proposed platform.

2. Implement more algorithms

Four congestion control algorithms are implemented on the proposed platform. However, many other algorithms including the single path algorithms have been proposed recently, such as TCP-TRIM [48], A^{2}DTCP [49], FSQCN [50], TaTCP [51] and so on, it is interesting to compare their performances with the implemented multipath algorithms. Thus, more algorithms need to be implemented on the proposed platform in the further.

Though being adopted by many data center and wireless network environment, MPTCP is not mature enough, and still has problems to be solved. The future research should be focused at these points:

(1) Congestion control

The problems existing in single-path congestion control are aggravated with the use of multiple paths, which is of great importance to be addressed with. These key challenges are mentioned below.

Heterogeneity. The Internet is composed of a vast variety of heterogeneous links and paths. Those paths have diverse features such as bandwidths, delay and packet loss rate, which changes with time and traffic loads. The design of congestion control algorithms that deal with this vast range of heterogeneity in a stable and efficient way is a challenging task.

Stability. The modeling of realistic network for stability analysis can be extremely complicated if packet sizes and heterogeneous RTT are taken into consideration. Usually, when doing simulations, it tends to use simple models and study complex behavior. However, due to the simplifying assumption in simulations, a mechanism that is found to be stable in simulations may be unstable in reality. Therefore, models that approximate reality more closely are desired.

Friendliness vs Responsiveness. Future research about a more efficient trade-off between responsiveness and friendliness is required.

(2) Energy consumption

Power consumption with the use of multiple paths is very high, which becomes a problem for mobile terminal. Related studies are conducted to make MPTCP more energy-friendly and energy efficient both in wireless network and data center environment. For popularizing MPTCP, energy consumption is a further research direction worthy of more attention.

(3) Internetworking with Middleboxes

Many middleboxes are widely deployed in today's Internet, enterprise and cellular networks to meet the need for more sophisticated functions, which need to be considered while the network protocol changes. Without considering the middleboxes effects, most current multipath protocol assume that data packet can be delivered to the destination without any modification, and this will bring problems when deploying multipath transport protocol. MPTCP handles middleboxes by reverting back to TCP in case of a conflict. Although MPTCP is capable of handling most of the middleboxes, some cases do exist that can degrade the performance of MPTCP. Consequently, it is of great significance to consider internetworking with middleboxes when deploying MPTCP.

(4) Cross-Layer Multipath Interactions

The application layer and the transport layer can work together to achieve better real-time performance. The transport layer and the network layer can work together to acquire minimizing congestion and differentiating between losses due to transmission or congestion in wireless network. Due to these benefits, cross-layer interaction has become a hot area of research.

Acknowledges

This work is supported by the National Natural Science Foundation of China under grant no. 61232001, 61572530.

References

[1] X.-w. Zheng, B. Hu, D.-j. Lu, H. Liu, A Multi-Objective Virtual Network Embedding Algorithm in Cloud Computing, *Journal of Internet Technology*, Vol. 17, No. 4, pp. 633-642, July, 2016.

[2] X. Zhang, Z. Tan, S. Xu, C. Tao, Utility Maximization based on Cross-layer Design for Multi-service in Macro-femto Heterogeneous Networks, *IEEE Transactions on Wireless Communications*, Vol. 12, No. 11, pp. 5607-5619, November, 2013.

[3] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, TCP Extensions for Multipath Operation with Multiple Addresses, *Internet Engineering Task Force*, RFC 6824, January, 2013.

[4] Y. Chen, D. Towsley, On Bufferbloat and Delay Analysis of Multipath TCP in Wireless Networks, *Networking Conference*, Trondheim, Norway, 2014, pp. 1-9.

[5] Y. Chen, Y. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, D. Towsley, A Measurement-based Study of Multipath TCP Performance over Wireless Networks, *Proceedings of the 2013 ACM Conference on Internet Measurement Conference*, Torre Telefonica, Barcelona, Spain, 2013, pp. 455-468.

[6] J. Gettys, Bufferbloat: Dark Buffers in the Internet, *IEEE Internet Computing*, Vol. 15, No. 3, pp. 96-96, May, 2011.

[7] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, J. Boudec, MPTCP is Not Pareto-optimal: Performance Issues and a Possible Solution, *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, Nice, France, 2012, pp. 1-12.

[8] Q. Peng, A. Walid, J. S. Hwang, S. H. Low, Multipath TCP: Analysis, Design, and Implementation, *IEEE/ACM Transactions on Networking*, Vol. 24, No. 1, pp. 596-609, February, 2016.

[9] T. A. Le, Improving the Performance of Multipath Congestion Control over Wireless Networks, *IEEE 2013 International Conference on Advanced Technologies for Communications (ATC)*, Ho Chi Minh, Vietnam, 2013, pp. 60-65.

[10] Y. Cao, M. Xu, X. Fu, Delay-based Congestion Control for Multipath TCP, *2012 20th IEEE International Conference on Network Protocols (ICNP)*, Austin, TX, 2012, pp. 1-10.

[11] X. Chen, R. Jin, K. Suh, B. Wang, W. Wei, Network

Performance of Smart Mobile Handhelds in a University Campus WiFi Network, *Proceedings of the 2012 ACM conference on Internet Measurement Conference*, Boston, MA, 2012, pp. 315-328.

[12] P. Dong, J. Wang, J. Huang, H. Wang, G. Min, Performance Enhancement of Multipath TCP for Wireless Communications With Multiple Radio Interfaces, *IEEE Transactions on Communications*, Vol. 64, No. 8, pp. 3456-3466, August, 2016.

[13] M. Zhang, J. Lai, A. Krishnamurthy, L. Peterson, R. Wang, A Transport Layer Approach for Improving End-to-End Performance and Robustness, *USENIX Annual Technical Conference*, Boston, MA, 2004, pp. 99-112.

[14] M. Honda, Y. Nishida, L. Eggert, P. Sarolahti, H. Tokuda, Multipath Congestion Control for Shared Bottleneck, *Proc. PFLDNeT workshop*, Tokyo, Japan, 2009, pp. 19-24.

[15] C. Raiciu, M. Handly, D. Wischik, Coupled Congestion Control for Multipath Transport Protocols, *Internet Engineering Task Force*, RFC6356, October, 2011.

[16] D. Wischik, C. Raiciu, A. Greenhalgh, M. Handley, Design, Implementation and Evaluation of Congestion Control for Multipath TCP, *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, 2011, pp. 99-102.

[17] T. A. Le, C. S. Hong, E. N. Huh, Coordinated TCP Westwood Congestion Control for Multiple Paths over Wireless Networks, *2012 International Conference on Information Networking (ICOIN)*, Bali, Indonesia, 2012, pp. 92-96.

[18] S. Ferlin-Oliveira, T. Dreibholz, O. Alay, Tackling the Challenge of Bufferbloat in Multi-Path Transport over Heterogeneous Wireless Networks, *2014 IEEE 22nd International Symposium of Quality of Service*, Hong Kong, China, 2014, pp. 123-128.

[19] K. Ma, J. He, Z. Zheng, X. Zeng, L. Fan, An Adaptive Congestion Control Algorithm Based on MPTCP, *Mobile Communications*, Vol. 37, No. 20, pp. 55-60, October, 2013.

[20] H. Jiang, Y. Wang, K. Lee, I. Rhee, Tackling Bufferbloat in 3G/4G Networks, *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, Boston, MA, 2012, pp. 329-342.

[21] H. Im, C. Joo, T. Lee, S. Bahk, Receiver-side TCP Countermeasure to Bufferbloat in Wireless Access Networks, *IEEE Transactions on Mobile Computing*, Vol. 15, No. 8, pp. 2080-2093, August, 2016.

[22] J. S. Lee, J. Lee, Multipath TCP Performance Improvement in Mobile Network, *2015 Seventh International Conference on Ubiquitous and Future Networks (ICUFN)*, Sapporo, Japan, 2015, pp. 710-714.

[23] X. Liu, F. Ren, R. Shu, T. Zhang, T. Dai, Mitigating Bufferbloat with Receiver-based TCP Flow Control Mechanism in Cellular Networks, *The 24th International Conference on Computer Communication and Networks*, Las Vegas, Nevada, 2015, pp. 1-8.

[24] K. Liu, J. Lee, On Improving TCP Performance Over Mobile Data Networks, *IEEE Transactions on Mobile Computing*, Vol. 15, No. 10, pp. 2522-2536, October, 2016.

[25] T. Dreibholz, M. Becke, E. P. Rathgeb, M. Tuxen, On the Use of Concurrent Multipath Transfer over Asymmetric Paths, *Global Telecommunications Conference (GLOBECOM 2010)*, Miami, FL, 2010, pp. 1-6.

[26] H. Adhari, T. Dreibholz, M. Becke, E. P. Rathgeb, M. Tuxen, Evaluation of Concurrent Multipath Transfer over Dissimilar Paths, *2011 25th International Conference on Advanced Information Networking and Applications (WAINA) Workshops,* Biopolis, Singapore, 2011, pp. 708-714.

[27] K. Liu, J. Y. B. Lee, Achieving High Throughput and Low Delay in Mobile Data Networks by Accurately Predicting Queue Lengths, *Proceedings of the 12th ACM International Conference on Computing Frontiers*, Ischia, Italy, 2015, Article No. 20.

[28] T. Osuga, K. Nogami, A Method for Correcting Minimum RTT of TCP Congestion Control in Mobile Networks, *IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, 2015, pp. 310-315.

[29] T. Dreibholz, E. P. Rathgeb, R. Seggelmann, M. Tüxen, Transmission Scheduling Optimizations for Concurrent Multipath Transfer, *Proceedings of the 8th International Workshop on Protocols for Future, Large-Scale and Diverse Network Transports~(PFLDNeT)*, Lancaster, Pennsylvania, 2010, pp. 1-6.

[30] C. Paasch, S. Ferlin, O. Alay, O. Bonaventure, Experimental Evaluation of Multipath TCP Schedulers, *Proceedings of the ACM SIGCOMM Workshop on Capacity Sharing Workshop*, Chicago, IL, 2014, pp. 27-32.

[31] N. Kuhn, E. Lochin, A. Mifdaoui, G. Sarwar, O. Mehani, R. Boreli, Daps: Intelligent Delay-aware Packet Scheduling for Multipath Transport, *IEEE International Conference on Communications (ICC)*, Sydney, Australia, 2014, pp. 1222-1227.

[32] F. Yang, Q. Wang, P. D. Amer, Out-of-Order Transmission for In-Order Arrival Scheduling for Multipath TCP, *2014 28th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, Victoria, Canada, 2014, pp. 749-752.

[33] Z. Ge, H. Liu, T. Li, Algorithm of Path Selection based on Kalman Filter for Mobile MPTCP, *Journal of Guangxi University (Natural Science Edition)*, Vol. 40, No. 5, pp. 1260-1266, October, 2015.

[34] T. Le, L. X. Bui, Forward Delay-based Packet Scheduling Algorithm for Multipath TCP, *Mobile Networks and Applications*, Vol. 23, No. 1, pp. 4-12, February, 2018.

[35] D. Zhou, W. Song, M. Shi, Goodput Improvement for Multipath TCP by Congestion Window Adaptation in Multi-radio Devices, *IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, 2013, pp. 508-514.

[36] C. Pluntke, L. Eggert, N. Kiukkonen, Saving Mobile Device Energy with Multipath TCP, *Proceedings of the sixth International Workshop on MobiArch*, Bethesda, MD, 2011, pp. 1-6.

[37] Y. Lim, Y. C. Chen, E. M. Nahum, D. Towsley. R. J. Gibbens,

How Green is Multipath TCP for Mobile Devices? *Proceedings of the 4th Workshop on All things Cellular: Operations, Applications, & Challenges*, Chicago, IL, 2014, pp. 3-8.

[38] Q. Peng, M. Chen, A. Walid, S. Low, Energy Efficient Multipath TCP for Mobile Devices, *Proceedings of the 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Philadelphia, PA, 2014, pp. 257-266.

[39] M. Li, A. Lukyanenko, S. Tarkoma, Y. Cui, A. Ylä-Jääski, Tolerating Path Heterogeneity in Multipath TCP with Bounded Receive Buffers, *SIGMETRICS*, Pittsburgh, PA, 2014, pp. 375-376.

[40] Z. Xia, Z. Chen, H. Ye, M. Zhao, An Improved MPTCP in Coded Wireless Mesh Networks, *2nd IEEE International Conference on Broadband Network & Multimedia Technology*, Beijing, China, 2009, pp. 795-799.

[41] M. Li, A. Lukyanenko, Y. Cui, Network Coding based Multipath TCP, *2012 IEEE Conference on Computer Communications Workshops (INFOCOM Workshops)*, Orlando, FL, 2012, pp. 25-30.

[42] Y. Cui, L. Wang, X. Wang, H. Wang, Y. Wang, FMTCP: A Fountain Code-based Multipath Transmission Control Protocol, *IEEE/ACM Transactions on Networking*, Vol. 23, No. 2, pp. 465-478, April, 2015.

[43] S. He, K. Xie, D. Zhang, Completion Time-Aware Flow Scheduling in Heterogenous Networks, *Algorithms and Architectures for Parallel Processing*, 2015, Zhangjiajie, China, pp. 492-507.

[44] Y. Cao, M. Xu, X. Fu, E. Dong, Explicit Multipath Congestion Control for Data Center Networks, *Proceedings of the 9th ACM Conference on Emerging Networking Experiments and Technologies*, Santa Barbara, CA, 2013, pp. 73-84.

[45] P. Mafole, Y. Manabe, T. Kitasuka, M. Aritsugi, Energy-Efficient Fragmentation Scheme for IEEE 802.11 DCF: Delay and Fairness Considerations, *Journal of Internet Technology*, Vol. 15, No. 4, pp. 625-634, July, 2014.

[46] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, J. L. Boudec, MPTCP is Not Pareto-optimal: Performance Issues and A Possible Solution, *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies,* Nice, France, 2012, pp. 1-12.

[47] B. Chihani, D. Collange, A Multipath TCP Model for ns-3 Simulator, *arXiv preprint* arXiv:1112.1932, December, 2011.

[48] T. Zhang, J. Wang, J. Huang, G. Min, Tuning the Aggressive TCP Behavior for Highly Concurrent HTTP Connections in Intra-Datacenter, *IEEE/ACM Transactions on Networking*, Vol. 25, No. 6, pp. 1-15, October, 2017.

[49] T. Zhang, J. Wang, J. Huang, Y. Huang, J. Chen, Y. Pan, Adaptive-Acceleration Data Center TCP, IEEE Transactions on Computers, Vol. 64, No. 6, pp. 1522-1533, January, 2015.

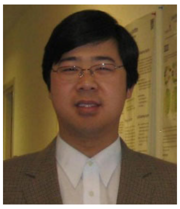
[50] C. Ruan, J. Wang, J. Wan, Y. Pan, FSQCN: Fast and Simple Quantized Congestion Notification in Data Center Ethernet, *Journal of Network and Computer Applications*, Vol. 83, No. C, pp. 53-62, April, 2017.

[51] S. Liu, J. Huang, Y. Zhou, J. Wang, T. He, Task-aware TCP in Data Center Networks, *Proceedings of the IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, 2017, pp. 1356-1366.

Biographies

Xuedong Sun is a master in School of Central South University. Her main research interests include computer network and WANs optimization.

Jianxin Wang is the vice dean and a professor in School of Information Science and Engineering, Central South University. His current research interests include algorithm analysis and optimization, parameterized algorithm, bioinformatics and computer network. He has published more than 200 papers in various International journals and refereed conferences.



Neal N. Xiong is current a faculty at Dept. of Business and Computer Science (BCS). His research interests include Cloud Computing, Business Networks, Security and Dependability, Parallel and Distributed Computing, and Optimization Theory. He has published over 100 international journal papers and over 100 international conference papers.

Pingping Dong received the B.S., M.S. and Ph.D. degrees from Central South University. She is currently a Teacher with the Department of Computer Education, Hunan Normal University, Changsha, China. Her research interests are focused on protocol optimization and protocol design in wide area networks and wireless local area networks.