# Enhanced Multipath QUIC Protocol with Lower Path Delay and Packet Loss Rate

Chih-Lin Hu[1], Fang-Yi Lin[1], Wu-Min Sung[1], Nien-Tzu Hsieh[1], Yung-Hui Chen[2], and Lin Hui[3]

[1] Department of Communication Engineering, National Central University, Taoyuan City 320317, Taiwan
[2] Department of Computer Information and Network Engineering, Lunghwa University of Science and Technology, Taoyuan City 333326, Taiwan
[3] Department of Computer Science and Information Engineering, Tamkang University, New Taipei City 25137, Taiwan
clhu@ce.ncu.edu.tw; fangyi.lin, wumin.sung, neintzu.hsieh@g.ncu.edu.tw; cyh@mail.lhu.edu.tw; 121678@mail.tku.edu.tw

**Abstract.** Ubiquitous content distribution brings enormous data flows on the internet, which overwhelms network services and applications. To mitigate this circumstance, the Quick UDP Internet Connect (QUIC) protocol can provide applications with flow-controlled streams for structured communication, low-latency connection establishment, and network path migration. Consider the high dynamics of traffic loading and resource provision on network hosts that forward data flows along a particular path between two endpoints. For faster media transfer, QUIC performs better than TCP for its effects in shortening the time of connection establishment and data transmission between two endpoints. Notwithstanding, the performance of QUIC is still susceptible to the bandwidth capacity of a single path and its variance. Recent studies attempted to exploit the notion of multipath QUIC that forwards the data over multi paths to not only augment the total bandwidth capacity but also avoid traffic congestion on some paths. Although prior studies showed the multipath effect, their QUIC schemes send out the data merely depending on the round-robin or the shortest-time-first scheduling between multiple paths. However, these ordinary designs cause the link congestion or high transmission cost within multiple paths. In this paper, our study proposes a novel multipath QUIC scheme which is able to minimize the flow completion time of multipath QUIC by jointly utilizing two measures of path delay and packet loss rate on a path. Experimental results show that the proposed algorithm is superior to other scheduling schemes, including QUIC, Lowest-RTT-First (LRF) QUIC, and Pluginized QUIC (PQUIC).

**Keywords:** Quick UDP Internet Connect (QUIC), Multipath Transport, HTTP, Content Distribution, Internet Protocol, Internet Services

## 1 Introduction

The HTTP protocol family [1] is the basis for global internet data communications, enabling the rapid development of Web browsers and internet services. HTTP/1.1 and HTTP/2 are two major web protocols. With the proliferation of user demands and mobile services, particularly mobile media streaming and AR/VR flows to an increasing user population, the functions provided by HTTP/1.1 and 2 are no longer sufficient. In 2013, the IETF organization proposed the RFC 9000, i.e., Quick UDP Internet Connect (QUIC) – a UDP-based multiplexed and secure transport protocol. QUIC is often known as the transport layer for HTTP/3. It is recommended to develop HTTP/3 with QUIC and UDP in place of conventional HTTP/1.1 and 2 with TCP or UDP for internet services and applications in wireless and mobile environments.

QUIC provides applications with flow-controlled streams for encrypted, multiplexed and reliable communication, low-latency connection establishment, and network path migration. It can sustain high dynamics of traffic loading and resource provision on network hosts, rather than HTTP/2 based on TCP, TLS 1.2, and other HTTP derivatives. Compared with TCP, QUIC need not the 3-way handshake mechanism, so it can greatly reduce the time of network connection establishment and transmission latency. With multiplexing and path migration, it can strengthen the control of congested networks, making it more suitable for emerging mobile services in Wi-Fi and 4G/5G environments.

Prior studies argued that the performance of QUIC can be affected in the case of delivering large-size data between two endpoints [3]. This is because the packet pacing policy is basically used to vary the transmission speed of each stream when numerous packets enter that stream. The overall completion time of a data flow in a stream will vary as well. Thus, data throughput of each flow through a link may not reach to the full bandwidth capacity. Moreover, internet operators may operate any self-protection controls by limiting the transmission rate of UDP flows, which avoids unpredictable threats to the system.

As Section 2 will review, recent studies used Multipath QUIC (MPQUIC) to deal with the above concerns subject to the restriction of a single path. Similar to Multipath TCP (MPTCP) [2], MPQUIC sends data through different paths and uses the aggregate bandwidth of different paths. It also likely modifies the *path scheduler* policy for increasing the transmission speed and thence decreasing the path delay that definitely corresponds to the end-to-end transmission delay of a QUIC stream between two endpoints in a network. In this paper, our study proposes a novel MPQUIC scheme which aims to reduce both path delay and packet loss rate simultaneously. The proposed MPQUIC scheme is able to obtain a shorter flow completion time for each stream in the network. We investigate the proposed MPQUIC scheme in comparison with QUIC, LRF, and PQUIC scheduling schemes. Experiments are driven using the Mininet emulator and the Abilene topology. Performance results show that the stable and efficient effects of our proposed scheme as demonstrated by the cumulative distribution function (CDF) of flow completion time. In addition, our proposed scheme obtains lower path delay and packet loss rate than the other schemes.

The rest of this paper is organized as follows. Section 2 describes related work. Section **??** details the problem formulation and the path selection algorithm. Section 4 describes the relative performance. Finally, the conclusion is given in Section 5.

## 2 Related Work

Inspired by MPTCP, [4] examined the concept of MPQUIC which arranges QUIC connections to go on different paths according to network characteristics. There are two main reasons for the use of the multi-path function. The first is to collect the network resource of different paths to transmit data. Automatically selecting the best path becomes an interesting idea. The second is to maintain user experience against network failure. Given a device with multiple ports, if one of the network interfaces/ports/paths fails, immediately switching to another one will not affect the user experience. Hence, using multi-path designs can ensure the reliability and stability of network transport services by distributing and scheduling streams to reduce the overall completion time in a network.

Recent studies proposed several MPQUIC scheduling methods. In [5], an environment-aware MPQUIC packet scheduling method was proposed to perform collaborative scheduling for optimize the overall system transmission time through the Round-Robin (RR) manner in sequential cycles. [6] emphasized on the fair allocation of aggregate bandwidth based on stream priority, thereby avoiding the delay of any individual stream due to heterogeneous paths. [7] showed a PStream scheduling design used to select paths to different streams according to the match of stream and path characteristics, unlike a greedy manner that all streams compete for the fast path. [8] developed a Priority Bucket method, which divides streams into different buckets according to stream priority. The priority and size factors can be extracted from HTTP/2 expression. When streams with the same priority exist in the same bucket, they are served in first-come-first-served order. In [9], the Peekaboo method based on reinforcement learning was proposed. It decided on a scheduling sequence by referring to the properties of temporal certainty and randomness of current path characteristics. [10] proposed a Nine Tails scheduler that can selectively use redundant paths to reduce latency as sending data in the tail part. By switching between redundant and non-redundant scheduling policies, it can have higher overall throughput and loss recovery. [11] developed a Pluginized QUIC (PQUIC) architecture which can modularize QUIC functions and load in a portion or full of QUIC modules. Through dynamic module deployment, the QUIC protocol was launched to improve the transmission performance.

The above literature review shows that previous studies on MPQUIC mainly used the QUIC-default Round-Trip Time (RTT) to determine the path selection. Our study considers two network-oriented factors, i.e., delay and packet loss rate of a path. Accordingly, we formulate a weighting normalization method to calculate the weights of paths, which can be used to facilitate path selection and thus minimize the flow completion time over MPQUIC streams.

## 3 Design of Path Selection Scheme

Give a network topology $G(V, L)$. For every link $l_{i,j} \in L$ from $v_i$ to $v_j$, the available bandwidth, the delay of the link, and the packet loss rate w.r.t $l_{i,j}$ are denoted as $b_{i,j}$, $t_{i,j}$ and $o_{i,j}$, respectively. Then, $b_{i,j}^{max}$ denotes the maximum amount of bandwidth that $l_{i,j}$ can use.

Let $F$ contain a set of all streams in $G(V, L)$, $\mathcal{P}_f^*$ represent a multipath set in use for a stream $f \in F$, $P_f^*[m]$ be the set of links in the $m^{th}$ path, and likewise $\mathcal{P}_f^*[m][n]$ be the $n^{th}$ link of the $m^{th}$ path. Thus, for the stream and path selection, we take $x_{l_{i,j}}^f$ to be a binary indicator, defined as follows:

$$x_{i,j}^f = \begin{cases} 1, & \text{if a stream } f \text{ passes through a link } l_{i,j} \ , \\ 0, & \text{other conditions.} \end{cases} \tag{1}$$

We further define several expressions regarding the relationship between links and paths, as follows:

$$b_f^P = min\Big(b_{i,j} \times x_{i,j}^f\Big), \quad \forall l \in l_{i,j},\ x_{i,j}^f \neq 0,\ f \in F \tag{2}$$

$$b_{i,j}^{max} \geq \sum_{f \in F} b_{i,j} \times x_{i,j}^f, \quad \forall l \in l_{i,j} \tag{3}$$

$$t_f^P = \sum_{l_{i,j} \in L} t_{i,j} \times x_{i,j}^f, \quad \forall\ f \in F,\ l \in l_{i,j} \tag{4}$$

$$o_f^P = 1 - \prod_{l_{i,j} \in L} \Big(1 - o_{i,j} \times x_{i,j}^f\Big), \quad \forall l \in l_{i,j},\ x_{i,j}^f \neq 0,\ f \in F \tag{5}$$

$$y(\mathcal{P}_f^*) = \begin{cases} 1, & \bigcup P_f^* \neq \emptyset, \\ 0, & \bigcup P_f^* = \emptyset. \end{cases} \tag{6}$$

Formula (2) indicates the available bandwidth of a stream $f$ in the set of paths $P$, and then takes the minimum value. (3) indicates that the bandwidth passed by a link cannot be greater than the maximum bandwidth available of the link. (4) means the sum of transmission delays on a link w.r.t. a stream $f$. (5) is to multiply the successful rate of each link to get the overall successful rate on a path, so as to obtain the packet loss rate of this path.

To transform a single-path stream into a multipath stream by (6), $y(\mathcal{P}_f^*)$ indicates whether any link and path in the set of paths $\mathcal{P}_f^*$ can be reused or not. Here, we further discuss two cases, as follows.

**Case 1** When the links and paths in $\mathcal{P}_f^*$ are not reused.

Since links are not reused, the sum of the available bandwidth of each path can be calculated by (7). Then, for $y(\mathcal{P}_f^*) = 0$ and $\forall v_j \in V$, we can formulate (8) to check the link condition of $v_i$ and $v_j$: (i) the total number of positive multipaths, (ii) the total number of negative multipaths, and (iii) a balanced state if both $v_i$ and $v_j$ are intermediate relays.

$$b_f^* = \sum_{P \in \mathcal{P}_f^*} b_f^P, \quad \forall f \in F,\ y(\mathcal{P}_f^*) = 0 \tag{7}$$

$$\sum_{l_{i,j} \in L} x_{i,j}^f - \sum_{l_{j,i} \in L} x_{j,i}^f = \begin{cases} |\mathcal{P}_f^*|, & \text{if } v_i \text{ is a start point of } f\ , \\ -|\mathcal{P}_f^*|, & \text{if } v_i \text{ is a target point of } f, \\ 0, & \text{if } v_i \text{ is a relay point of } f. \end{cases} \tag{8}$$

□

**Case 2** When the links and paths in $\mathcal{P}_f^*$ can be reused

Let $z_{l_{i,j}}^{\mathcal{P}_f^*}$ indicate whether $l_{i,j}$ is reused in $\mathcal{P}_f^*$:

$$z_{l_{i,j}}^{P_f^*} = \begin{cases} 1, & l_{i,j} \subseteq \bigcup \mathcal{P}_f^*, \\ 0, & l_{i,j} \not\subseteq \bigcup \mathcal{P}_f^*. \end{cases} \tag{9}$$

$n(l_{i,j}, P_f^*)$ indicates the number of times that $l_{i,j}$ is reused by some paths in $\mathcal{P}_f^*$:

$$n(l_{i,j}, P_f^*) = \begin{cases} \sum\limits_{m \in |\mathcal{P}_f^*|} \sum\limits_{n \in |\mathcal{P}_f^*[m]|} l_{i,j} \wedge P_f^*[m][n] - 1, & \forall f \in F,\ l_{i,j} \in L,\ z_{l_{i,j}}^{P_i^*} = 1, \\ 0, & \forall f \in F,\ l_{i,j} \in L,\ z_{l_{i,j}}^{P_i^*} = 0. \end{cases} \tag{10}$$

Then, the bandwidth of a link is divided into two parts: the link bandwidth that has been reused $\bar{b_f^*}$, and the link that has not been reused $\hat{b_f^*}$, as follows.

$$b_f^* = \bar{b_f^*} + \hat{b_f^*}, \quad \forall f \in F, \quad \text{subject to} \tag{11a}$$

$$\bar{b_f^*} = min(b_f^P), \quad \forall f \in F,\ P \in \mathcal{P}_f^*,\ y(\mathcal{P}_f^*) = 1,\ z_{l_{i,j}}^{P_i^*} = 1. \tag{11b}$$

$$\hat{b_f^*} = \sum_{P \in \mathcal{P}_f^*} b_f^P, \quad \forall f \in F,\ y(\mathcal{P}_f^*) = 1,\ z_{l_{i,j}}^{P_i^*} = 0, \tag{11c}$$

Formula (11a) adds the two parts together, which yields the total amount of bandwidth that a path set can provide.

Formula (12) clarifies the link relation in three conditions. (i) If $v_i$ is a start point of a stream $f$, the total of paths that a steam can still use is given by $|\mathcal{P}_f^*|$ minus the number of times $l_{i,j}$ that is currently used by some paths in $\mathcal{P}_f^*$, i.e., $n(l_{i,j}, P_f^*)$. (ii) If $v_i$ is a target point, the calculation is in opposition to (i). (iii) Finally, if $v_i$ is a relay w.r.t. $\forall\ y(\mathcal{P}_f^*) = 1$ and $v_j \in V$, there are three sub-cases (a)(b)(c). Explicitly, (a) multiple paths converge at this relay point, then $n(l_{j,i}, P_f^*) - n(l_{i,j}, P_f^*)$ is negative. (b) multiple paths to divert from this point, this outcome is positive. (c) in a balanced state, the outcome equals to 0.

$$\sum_{l_{i,j} \in L} x_{l_{i,j}}^f - \sum_{l_{j,i} \in L} x_{l_{j,i}}^f = \begin{cases} |\mathcal{P}_f^*| - n(l_{i,j}, P_f^*), & \text{if } v_i \text{ is a start point of } f\ , \\ -|\mathcal{P}_f^*| + n(l_{j,i}, P_f^*), & \text{if } v_i \text{ is a target point of } f, \\ n(l_{j,i}, P_f^*) - n(l_{i,j}, P_f^*), & \text{if } v_i \text{ is a relay point of } f. \end{cases} \tag{12}$$

□

Note that under the multipath scenario, the delay time and packet loss rate of a path are not affected by whether a path is reused subject to (2). Regardless of the value of (6), the delay time and packet loss rate w.r.t. any $P \in \mathcal{P}_f^*$, denoted as $t_f^*$ and $o_f^*$, can be given below.

$$t_f^* = \max(t_f^P) \ , \quad \forall f \in F, \ P \in \mathcal{P}_f^*, \ y(\mathcal{P}_f^*) = 0 \tag{13}$$

$$o_f^* = \sum_{P \in \mathcal{P}_f^n} \frac{o_f^P}{|\mathcal{P}_f^*|}, \quad \forall \ f \in F, y(\mathcal{P}_f^*) = 0 \tag{14}$$

According to (13), given a set of final selected multipaths, the delay time is represented by the maximum delay time on the path for $\forall P \in \mathcal{P}_f^*$. The outcome of (14) indicates the average of packet loss rate for those selected paths in $\mathcal{P}_f^*$. After calculating the available bandwidth, delay time, and packet loss rate, now, it is able to figure out the comparison between user requirements and actually available provision, as explained below.:

$$b_f \leq b_f^*, \quad \forall f \in F \tag{15}$$

$$t_f \geq t_f^*, \quad \forall f \in F \tag{16}$$

$$o_f \geq o_f^*, \quad \forall f \in F \tag{17}$$

Particularly, (15) ensures that the multipath bandwidth is available for streaming $f$, while (16) and (17) enforce that both transmission delay and packet loss rate in the selected path need to be smaller than the tolerable bounds as requested by $f$.

Hence, in accordance with the above formulae and constraints of the multipath provision, we develop an optimal multipath selection problem of minimizing the flow completion time subject to user requirements, as expressed below:

$$\begin{array}{ll} \arg\ \min \sum_{f \in F} t_f^*, & \\ \text{s.t.} & \\ \quad x_{i,j}^f = 1, & \forall l_{i,j} \in L, \\ \quad z_{l_{i,j}}^{\mathcal{P}_f^*} \in (0,1), & \forall \mathcal{P}_f^*, \ l_{i,j} \in L, \\ \quad y(\mathcal{P}_f^*) \in (0,1), & \forall \mathcal{P}_f^* \in \mathcal{P}, \\ \quad \text{Eqs. } (15), (16), (17). & \end{array} \tag{18}$$

Our study refers to the research efforts in [12][13], and learns that such a multipath selection problem for QoS-based data streaming is known as NP-Complete [14]. Instead of finding a static optimization in theory, our study in this paper attempts to develop an optimal-approximate solution to figure out a set of appropriate multipaths using heuristic strategies with two design factors, i.e., path delay and packet loss rates. Particularly, we describe a weighting normalization method in 19 with two tuning parameters $\alpha$ and $\beta$ to change the relative influence of path delay and packet loss rate over MPQUIC streams.

$$p_w = \alpha \times \frac{t_f}{t_f^*} + \beta \times \frac{o_f}{o_f^*}. \tag{19}$$

In what follows, we specify the algorithmic procedures for finding the paths for MPQUIC streams.

**Algorithm 1** Path Set Formation with Joint Path Delay and Packet Loss Rate

When the stream enters the MPQUIC, the system initializes the set of available paths $\mathcal{P}_f$ for a data stream $f$, as well as prepares an empty two-dimensional matrix $A[\ ][\ ]$. At first, when $\mathcal{P}_f$ is empty, the system refers to (2), (4) and (5) to determine the values of data stream bandwidth, delay, and packet loss rate, which are stored in $A[\ ][\ ]$. Then, the system checks a condition of whether the set of available paths for $f$ contains equal to or more than $k$ paths. As this condition is valid, the system proceeds to Algorithm 2 with a set of candidate paths for $f$. Later soon, Algorithm 2 will figure out $k$ shortest paths to form a set of $\mathcal{P}_f^*$.

**Algorithm 1** Path Set Selection with Joint Path Delay and Packet Loss Rate

```
input  : $G(V, L)$: network topology,
         $k$: the number of paths in the multipath,
         $\alpha$: a coefficient of path delay,
         $\beta$: a coefficicient of packet loss.
output: $\mathcal{P}_f^*$: the set of multipath.
1  while Flow $f$ comes into the system do
2      $\mathcal{P}_f = \{\emptyset\}$;
       $A[\ ][\ ] = $ null;
       while $\mathcal{P}_f = \{\emptyset\}$ do
3          $\mathcal{P}_f \leftarrow getDefaultPathSet(\mathcal{P}, f)$ ;
           foreach $p \in \mathcal{P}_f$ do
4              $A[p][0] \leftarrow getPathBW(\mathcal{P}[p])$ ;            ▷ (2)
               $A[p][1] \leftarrow getPathDelay(\mathcal{P}[p])$ ;         ▷ (4)
               $A[p][2] \leftarrow getPathPL(\mathcal{P}[p])$ ;            ▷ (5)
5          end foreach
6      end while
7      if ($P_f = \{\emptyset\}$ or $|P_f| < k$) then
8          Reject $f$ ;
9      else
10         $\mathcal{P}_f^* \leftarrow getkPath(\mathcal{P}_f, \alpha, \beta, f, k, A)$ ;   ▷ Go to Alg. 2
           if $\mathcal{P}_f^* = \emptyset$ then
11             $\mathcal{P}_f^* \leftarrow getShorestkPath \in \mathcal{P}_f$ ;
12         end if
13     end if
14 end while
```

**Algorithm 2** Finding $k$ Shortest Paths over MPQUIC Streams

Algorithm 2 is the path selection procedure for finding the k-shortest paths based on QoS requirements. This procedure refers to Yen's k-shortest path algorithm [15] with QoS-specific conditions. To find the k-shortest paths, the procedure runs several routes sequentially: (a) define variables $p_w$, $b_f^*$ and $\mathcal{P}_f^*[\,][\,]$, (b) calculate the weight value $p_w$ of a stream by (19), (c) sort the weights of streams in descending order, and (d) update the available bandwidth of each link according to (7) and (11a). Then, the procedural routine goes into a while-loop with a condition as $b_f^*$ is smaller than the bandwidth $b_f$ asked by a stream $f$. If the minimum bandwidth of $\mathcal{P}_f^*$ exceeds the currently available path $\mathcal{P}_f$, $\mathcal{P}_f^*$ is still to be null. Then, the routine updates the set of available paths $\mathcal{P}_f^*$ and the bandwidth $b_f^*$, remove the path of the smaller bandwidth from $\mathcal{P}_f^*$, add a path with the larger bandwidth, update $b_f^*$, and then push the value of $\mathcal{P}_f^*$ back to Algorithm 1 to allocate available paths. Eventually, the data flow is passed through those suitable and multiple paths in the current network. To better explore the effects of Algorithms 1 and 2, we will present experiments and performance results in Section 4.

**Algorithm 2** Finding $k$ Shortest Paths over MPQUIC Streams

```
Function $getkPath(\mathcal{P}_f, \alpha, \beta, f, k, A)$ is
    $p_w[\,] = null;$
    $b_f^* = 0;$
    $\mathcal{P}_f^*[\,][\,] = null;$
    foreach $p \in \mathcal{P}_f$ do
        $p_w[p] \leftarrow getPathWeight(\mathcal{P}[p], \alpha, \beta, A)$          ▷ (19)
    end foreach
    $p_w \leftarrow sortByDescendingOrder(p_w)$ ;
    $\mathcal{P}_f^* \leftarrow selectPathTopk(p_w, k)$ ;
    $b_f^* \leftarrow getMultiPathBW(\mathcal{P}_f^*)$          ▷ (7) and (11a)
    while $b_f^* \leq b_f$ do
        if $minBWPath(\mathcal{P}_f^*) \geq maxBWPath(\mathcal{P}_f - \mathcal{P}_f^*)$ then
            $\mathcal{P}_f^* = \emptyset$
            break;
        end if
        $\mathcal{P}_f^* \leftarrow \mathcal{P}_f^* - minBWPath(\mathcal{P}_f^*)$
        $\mathcal{P}_f^* \leftarrow \mathcal{P}_f^* + maxBWPath(\mathcal{P}_f - \mathcal{P}_f^*)$
        $b_f^* \leftarrow getMultiPathBW(\mathcal{P}_f^*)$          ▷ (7) and (11a)
    end while
    return $\mathcal{P}_f^*$
end
```

## 4 Performance Results

This section shows the performance of our proposed method in comparison with QUIC, multipath QUIC LRF [16] and the PQUIC schemes [11].

### 4.1 Experimental Setting

We conducted experiments on the Mininet simulation platform that runs on a computer equipped with an Intel Core i7 processor, 16GB memory, and Ubuntu 18.04.6 LTS. All the algorithmic programs are coded in C language. We used the Wireshark packet analyzer to trace the data flows during the experiments.

Experiments were divided into three sorts with different sizes per data flow: 100, 200, and 400 MB, and produced three measure results of the overall flow completion time, path delay, and packet loss rate. We employed the Mininet to adjust simulation parameters. Explicitly, we set $k = 3$, delay coefficient $\alpha = 0.5$ and packet loss coefficient $\beta = 0.5$ as calculating the weighted value $p_w$. We adopted the Abilene topology [17]: there are 11 nodes and 14 links, the size of each packet is between 960 and 1200 bytes, the path bandwidth is set to 100 Mbps, the delay is from 0 to 100 ms by the binomial distribution, and packet loss rate is set to 0.001%. All experimental cases were run in 20 times to have the results on average.

### 4.2 Flow Completion Time

Figures 1, 2 and 3 exhibit the flow completion time in terms of the cumulative distribution function (CDF). As observed, the performance by naive QUIC is the worst, because QUIC only transmits data through a single path, as compared with the other schemes that take multiple paths. It is visible that our scheme outperforms LRF and PQUIC. Explicitly, LRF is based on finding the path with the minimum RTT for transmitting the top-priority data first. Thus, LRF behaves like a greedy way and only focuses on the RTT condition without referring to other network characteristics.

PQUIC switches between multipaths to ensure that data packets are sent to the receiver fairly. However, PQUIC suffers from minor performance degradation as path characteristics often change, and as the data size becomes larger. Relatively, our proposed scheme considers both path delay and packet loss rate of path candidates. By using a weighting normalization method, it is able to calculate $P_w$. The higher $P_w$, the higher priority the data needs to be scheduled for transmission first. Our proposed scheme with weighting effects can minimize the flow completion time, resulting in a remarkable comparison with LRF and PQUIC.

### 4.3 Packet Loss Rate

Figures 4, 5 and 6 present the packet loss rate of the overall system performance. As observed, the packet loss rate of QUIC is higher than the other multipath schemes, for the major reason that only the resource allocation of a single path is used. In the case of data size 100 MB per stream, the packet loss rates of QUIC, LRF, and PQUIC are similar, but become different when the data size per stream increases to 200 MB and even 400 MB. LRF searches for the path of the minimum RTT, which may cause the problem of packet loss in the rear tail

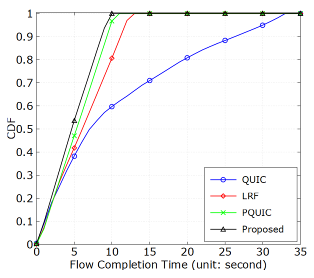

Fig. 1: Data size 100 MB

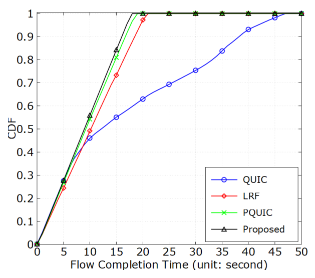

Fig. 2: Data size 200 MB

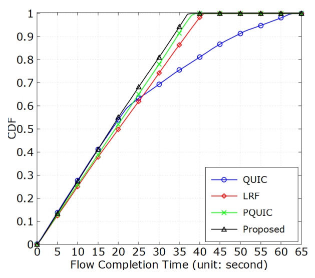

Fig. 3: Data size 400 MB

of data stream. PQUIC is fairer as allocating multiple paths to a data stream. Its packet loss rate is lower than the LRF's result. By contrast, our scheme can distribute the data to multiple paths efficiently, thereby being less susceptible to the increase of data size per stream. As seen, our scheme is able to cope with the packet loss rate to be lower than 1 % regardless the increasing data size from 100 MB to 400 MB.

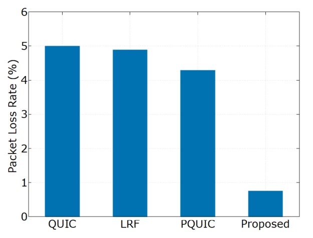

Fig. 4: Data size 100 MB

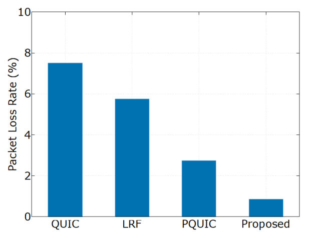

Fig. 5: Data size 200 MB

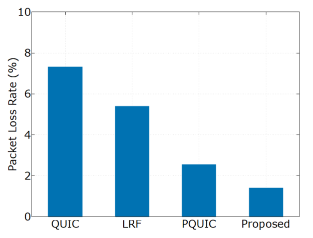

Fig. 6: Data size 400 MB

## 4.4 Overall system stability

Figures 7, 8 and 9 depict the quartile distribution of flow completion time when the experiment launched 20 data flows one by one repeatedly. Obviously, QUIC needs to take much more time to accomplish the transmission of per data flow. The time gap between QUIC and three multipath QUIC scheme is apparent. Instead, Figures 10, 11 and 12 exhibit a clear view on the time gap of three multipath QUIC schemes. LRF has not only a larger completion time but also a wider quartile distribution than PQUIC and our scheme. We examined that as compared with our scheme, PQUIC cannot perfectly allocate data packets to paths. As the amount of data packets increases rapidly, the probability of head-of-line blocking will increase and then affect the data throughput. Our scheme still keeps a minor quartile distribution with the lowest flow completion time, which shows the stable transport performance.

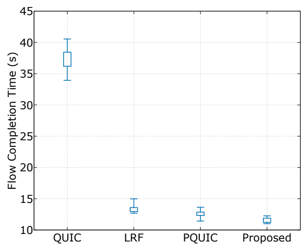

Fig. 7: Data size 100 MB

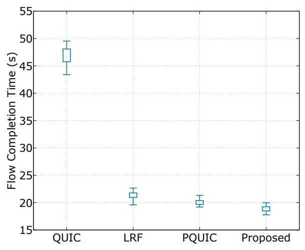

Fig. 8: Data size 200 MB

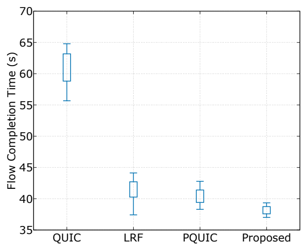

Fig. 9: Data size 400 MB

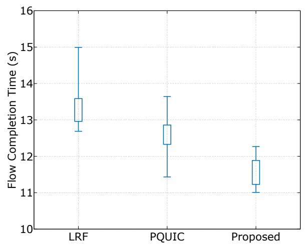

Fig. 10: Data size 100 MB

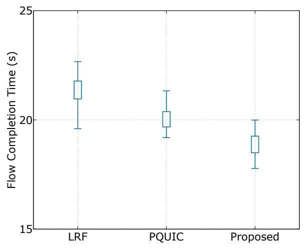

Fig. 11: Data size 200 MB

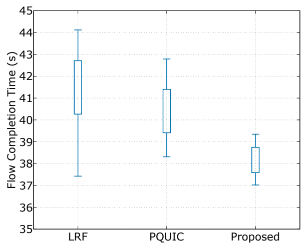

Fig. 12: Data size 400 MB

## 5 Conclusion

This paper designs a novel data transport scheme based on MPQUIC. Compared with the traditional network protocol TCP, MPQUIC is based on UDP and keeps the advantages of QUIC from a single-path to multi-path data transport. Our proposed MPQUIC scheme is able to joint sustain transmission delay and packet loss rate with respect to data flows. Performance study is conducted by comparing the proposed scheme with three prior schemes, i.e., QUIC, LRF, and PQUIC. It is remarkable that our proposed scheme performs efficiently and stably in terms of the flow completion time in the system. Our future research will try to implement MPQUIC and measure the transport performance in more complicated network scenarios with emerging AR/VR applications, particularly in mobile environments.

## Acknowledgment

This work was supported in part by the National Science and Technology Council, Taiwan (R.O.C.), under Contracts MOST-109-2221-E-008-051, NSTC-111-2221-E-008-064 and NSTC-111-2410-H-262-001.

## References

1. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, “Hypertext transfer protocol–http/1.1,” 1999.
2. C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley, “How hard can it be? designing and implementing a deployable multipath TCP,” In Proc. of 9th USENIX Symposium on Networked Systems Design and Implementation, Apr. 2012, pp. 399–412.
3. P. Megyesi, Z. Krämer, and S. Molnár, “How quick is quic?” In Proc. of 2016 IEEE International Conference on Communications (ICC’16), 2016, pp. 1–6.
4. Q. De Coninck and O. Bonaventure, “Multipath quic: design and evaluation,” In Proc. of the 13th international conference on emerging networking experiments and technologies, 2017, pp. 160–166.
5. Y. G. Jing Wang and C. Xu, “A stream-aware multipath quic scheduler for heterogeneous paths,” In Proc. of 2019 ACM 3rd Asia-Pacific Workshop on Networking, 2019, pp. 43–49.
6. A. Rabitsch, P. Hurtig, and A. Brunstrom, “A stream-aware multipath quic scheduler for heterogeneous paths,” In Proc. of of the Workshop on the Evolution, Performance, and Interoperability of QUIC, 2018, pp. 29–35.
7. X. Shi, L. Wang, F. Zhang, B. Zhou, and Z. Liu, “Pstream: Priority-based stream scheduling for heterogeneous paths in multipath-quic,” In Proc. of 29th International Conference on Computer Communications and Networks, 2020, pp. 1–8.
8. X. Shi, F. Zhang, and Z. Liu, “Prioritybucket: A multipath-quic scheduler on accelerating first rendering time in page loading,” In Proc. of the 11th ACM International Conference on Future Energy Systems, 2020, pp. 572–577.
9. H. Wu, Ã. Alay, A. Brunstrom, S. Ferlin, and G. Caso, “Peekaboo: Learning-based multipath scheduling for dynamic heterogeneous environments,” IEEE Journal on Selected Areas in Communications, vol. 38, no. 10, pp. 2295–2310, 2020.
10. V. A. Vu and B. Walker, “On the latency of multipath-quic in real-time applications,” In Proc. of 0 16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2020, pp. 1–7.
11. Q. De Coninck, F. Michel, M. Piraux, F. Rochet, T. Given-Wilson, A. Legay, O. Pereira, and O. Bonaventure, “Pluginizing quic,” In Proc. of the ACM Special Interest Group on Data Communication, 2019, pp. 59–74.
12. Z. Wang and J. Crowcroft, “Quality-of-service routing for supporting multimedia applications,” IEEE Journal on Selected Areas in Communications, vol. 14, no. 7, pp. 1228–1234, Sept. 1996.
13. C.-L. Hu, C.-Y. Hsu, and W.-M. Sung, “Fitpath: Qos-based path selection with fittingness measure in integrated edge computing and software-defined networks,” IEEE Access, vol. 10, pp. 45 576–45 593, 2022.
14. R. M. Karp, “Reducibility among combinatorial problems,” R. E. Miller and J. W. Thatcher (eds.) Complexity of Computer Computations. Boston, MA, USA: Plenum Press, 1972, ISBN 0-306-30707-3, pp. 85–103.
15. J. Y. Yen, “Finding the k shortest loopless paths in a network,” Management Science, vol. 17, no. 11, pp. 712–716, 1971.
16. T. Viernickel, A. Froemmgen, A. Rizk, B. Koldehofe, and R. Steinmetz, “Multipath quic: A deployable multipath transport protocol,” In Proc. of 2018 IEEE International Conference on Communications (ICC), 2018, pp. 1–7.
17. S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, “The internet topology zoo,” IEEE Journal on Selected Areas in Communications, vol. 29, no. 9, pp. 1765–1775, 2011.