Novel MultiPipe QUIC Protocols to Enhance the Wireless Network Performance

Gunjan Kumar Choudhary, Madhan Raj Kanagarathinam,
Harikrishnan Natarajan, Karthikeyan Arunachalam,
Sujith Regan Jayaseelan and Gaurav Sinha,
Samsung R&D Institute India Bangalore
Email: [gunjan.kc, madhan.raj, hari.n, karthikeya.a,
s.jayaseelan, g.sinha]@samsung.com

Debabrata Das
Networking & Communication Research Lab
IIIT Bangalore, India
ddas@iiitb.ac.in

***Abstract*—To improve the performance of the Transmission Control Protocol (TCP), the Quick UDP Internet Connections (QUIC) was introduced. However, from the recent literature, in the wireless networks, QUIC does not fully utilize the link capacity, because of varying network wireless medium characteristics for an application. To the best of our knowledge, for the first time, this paper has proposed two novel protocols to overcome the wireless network's challenges, by MultiPipe (multi sessions) QUIC protocol (MP-QUIC). The two MP-QUIC's novel protocols are (i) Round Robin MP-QUIC (RR-MP-QUIC) and (ii) Cross-Layer Burst Aware MP-QUIC (CBA-MP-QUIC). In RR-MP-QUIC, multiple pipes are created between source as well as destination and assign a pipe to each object in a round-robin manner. We set up the testbed over live air commercial network. The experimental results reveal that the RR-MP-QUIC Page Load Time (PLT) of a web page decreases by 76% with respect to QUIC. Furthermore, to improve the performance of RR-MP-QUIC, we proposed CBA-MP-QUIC, which optimally creates pipes, efficiently schedules the objects and understands with respect to cross-layer (wireless channel characteristics and load in data link layer) parameters. Hence this adapts dynamically to the network conditions and maintains fairness between the applications in User Equipment (UE) and within the same application. Experimental result of CBA-MP-QUIC on live air improves the web page PLT by 143% with respect to QUIC.**

***Index Terms*—QUIC, MP-QUIC, Pipelining, Page loading, Link Utilization, Multi Session, MultiPipe**

I. Introduction

HTTP/2 uses multiplexing pipelining, a method for sending multiple connections/streams/objects in one socket or pipe or session over the underlying transport layer as TCP. Its packets are delivered in-order at the receiver to the application, thus suffer from performance with TCP Head Of Line (HOL) blocking as mentioned in [1]. To overcome this Google proposed QUIC, a new data protocol, which replaces TCP and uses UDP at the transport layer. Hence does not have the three-way handshake provision leading to zero Round Trip Time (RTT) connection [2].

Though the initial motto of QUIC is for web applications, with the advent of better error correction mechanisms, it will be also used for live video streaming and download services as well. However, with the increase in demand for QUIC of Internet traffic [3], migration of the above-mentioned services to QUIC would be sooner. QUIC working group has been discussing the open challenges of multipath support, Forward Error Correction (FEC) and also network management issues.

Due to bursty and dynamic nature of the mobile data traffic, there would be sporadic packet losses caused due to high Bit Error Rate (BER) in the wireless access network (4G, 4G-Advanced and 5G) from network signal condition change, mobility, congestion in the path and hand-off problems, etc. On the other hand, leveraging multiple connections in lossy wireless network conditions overcomes the under-utilization of links. In [4], authors did performance evaluation of multiple HTTP connections over HTTP/2 in different packet loss conditions and showed that HTTP/2 performed inefficiently than the HTTP with multiple connections during lossy network conditions ($\gg 0.01\%$).

This motivated us to evaluate QUIC in various wireless channel conditions. We simulated different network conditions on an experimental testbed to compare QUIC with HTTP, HTTP/2, HTTP with three connections (mhttp C-3) and HTTP with ten connections (mhttp C-10). From the results in Fig. 1, we can infer that the legacy QUIC under-performs with respect to the HTTP with multiple connections as the packet loss increases.

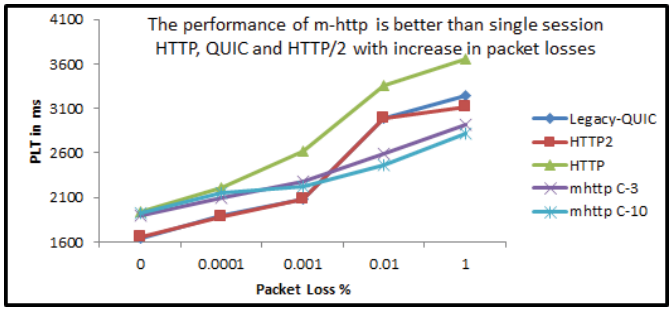

Fig. 1. PLT Comparison of different protocols under varied packet loss

Similarly, in [5], the authors proposed multiple QUIC schemes for improving the performance of web content delivery. The proposed solution tries to mitigate the uncertain fluctuations in RTT and random loss events at the wireless interface by retaining the resilient congestion window. This proposal requires modification in the server by estimating the congestion window for achieving performance and not all the client characteristics would be known. This motivated us to propose a client-only solution, by monitoring and evaluating the client-side wireless characteristics.

With respect to the above inevitable wireless network

978-1-7281-3106-1/20/$31.00 ©2020 IEEE

challenges, to the best of our knowledge for the first time, this paper has proposed two novel protocols for MP-QUIC [6], which are, Round-Robin MP-QUIC (RR-MP-QUIC) and Cross-Layer Burst Aware MP-QUIC (CBA-MP-QUIC). These protocols maximize the bandwidth utilization, and they are client-only solutions, which can be easily deployed without any changes in the network or at the server. These MP-QUIC protocols are further explained below

1) **RR-MP-QUIC**: protocol which creates multiple pipes (sessions) between the UE and the server, and assigns each pipe to an object (the home page of a website having multiple objects e.g. images, text, video, etc.; each one of these is one object.) request in a round-robin manner, more details are mentioned in Section III.
2) **CBA-MP-QUIC**: protocol which would create MultiPipe as in RR-MP-QUIC, but efficiently schedules the objects and understands the cross-layer (Signal-to-Interference-plus-Noise Ratio - SINR and load in traffic) parameters. Hence, adapting dynamically to the wireless network. This protocol provides intra-pipe fairness while maintaining scalability and it works on three newly introduced modules (new architecture) namely, Classifier, Path Manager and Session Creator, which are further explained in Section IV.

We compared MP-QUIC protocols in live air experiments in terms of PLT, in multiple locations, and infer that our approach outperforms the legacy QUIC. We found that RR-MP-QUIC performed 76% and CBA-MP-QUIC performed 143% better for page downloads than legacy QUIC.

The paper is organized as follows: Section II delineates an overview of the literature survey on the existing MP-QUIC solutions. Proposed MP-QUIC protocols are discussed in Section III. The proposed CBA-MP-QUIC system architecture is outlined in Section IV and the Algorithm 1 in Section V. Section VI provides the evaluation of live air results. Finally, concluding remarks are provided in Section VII.

II. Literature Survey

In [7], the authors investigated QUIC performance with SPDY and HTTP. The experimental results show that QUIC gives higher goodput with respect to TCP CUBIC and also outperforms SPDY in case of a lossy channel. However, the performance worsens during error conditions and FEC module enabled. In [8], the authors proposed a system called Parallel Streaming (ParS) which uses parallel connections to fetch each chunk of the video file using HTTP range requests. However, this proposal is specific to HTTP range requests and doesn't support small HTTP objects. [9], [10], and [11] proposed efficient network connectivity mechanisms to improve the page loading time in the Android system. But these works are limited to TCP and also cause redundant connections. In [12], the authors investigated the performance of adaptive bit-rate streaming algorithms (DASH) over QUIC. During experimental analysis, they observed that the degradation of streamed quality bit-rate with the use of QUIC affects overall QoE. In [13], the authors did compare the QUIC performance in different network conditions with other protocols HTTP/2 and HTTP/1.1. In [14], the authors dynamically adjust the Congestion Window based on the RTT update and network losses to efficiently utilize the bandwidth but is limited to TCP. In [15], authors compared the QUIC with TCP (where the underlying protocol is HTTP/2), they found that QUIC performs poorly when the packet losses and RTT are high.

All the recent literature mentioned above points that QUIC performance is not always better than other protocols in all network conditions with variable load. They do not solve the link utilization in the case of variable mobile network channel conditions as well as the fairness of sessions.

III. Proposed MultiPipe QUIC Protocols

A. Round Robin (RR) MP-QUIC

This protocol creates the pipes between the UE's application and the server as in the request comes as shown in Fig. 2. In Stage 1, the pipe is created and object1 is assigned to pipe in Stage 2. When a new request of object2 arrives then a new pipe is created at Stage 3 and object2 is assigned to the newly created pipe in Stage 4. The objects are scheduled in a round-robin manner as explained below.

The P_{max} is the maximum number of pipes that can be created, which is statically predefined. ID_{curr} is the current count of the object requests. ID_{obj} is the id of an incoming object, which will be assigned to some pipe. The ID_{obj} is incremented for each incoming object request.

$$ID_{obj} \leftarrow ID_{curr} + 1$$

Hence, the Pipe P_{id} for new incoming object is decided as below,

$$P_{id} = ID_{obj} \mod P_{max} \quad (1)$$

This protocol creates MultiPipes, when it is not required and does not adjust the number of pipes dynamically to the network conditions, leading to high resource utilization.

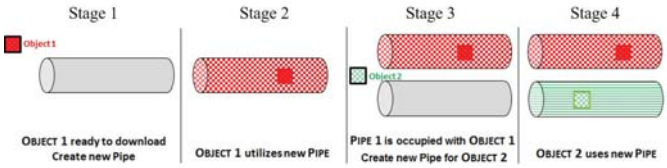

Fig. 2. RR-MP-QUIC Pipe Creation at UE

B. Shortcomings of RR-MP-QUIC

RR-MP-QUIC which schedules the object to pipe in a round-robin manner without any additional intelligence has the following limitations:

MultiPipe is not required always: In the case of good network conditions, even a single pipe is enough to process all the object requests. Also in case of a session with very few object requests or sessions with very less data requirements, a single pipe would still suffice.

Over or Under utilization of Pipe: The maximum number of pipes to be created and the object's scheduling order to

pipe is always fixed. This would end up in over-utilization or under-utilization of a particular pipe.

Intra or Inter Fairness Violations: Fairness would be impacted. This might include bandwidth stealing from users running legacy QUIC approach (Inter Fairness Violation) and also amongst the users with MultiPipe approach starting the download at a different time (Intra Fairness Violation).

Resource Utilization: The UE's power consumption, CPU and memory utilization for the RR-MP-QUIC approach would be relatively higher than the CBA-MP-QUIC as the former opens more pipes even in good network conditions.

To overcome the aforesaid limitation, CBA-MP-QUIC is designed in such a way that it is aware of the Cross-Layer information, schedules the objects efficiently and fair manner. Different from the wired environment, the wireless network conditions might share bottleneck links and hence suffers from various level of loss (such as burst loss, spurious wireless loss, handover loss, etc.) and also fluctuation in the signal level (such as SINR) and connection-level parameters (such as RTT, Packet Loss). The CBA-MP-QUIC runs on the client-side (in UE) understands the current wireless network conditions and works agnostically to the server-side implementations.

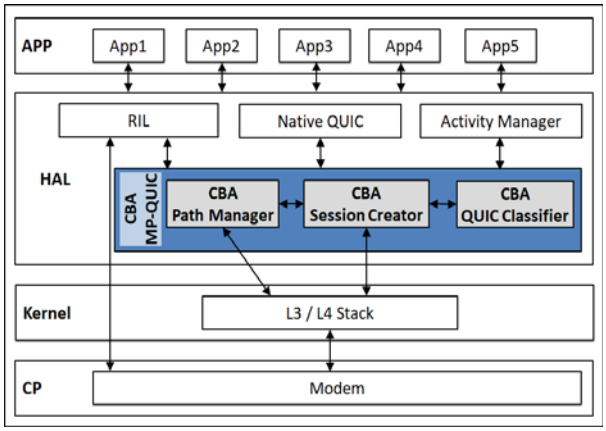

Fig. 3. Architecture of CBA-MP-QUIC

IV. CBA-MP-QUIC Proposed Architecture

Our proposed CBA-MP-QUIC architecture comprises of below described functional components as depicted in Fig. 3 (Blue coloured - CBA-MP-QUIC module).

A. CBA Path Manager

In this section, we provide an overview of information passed across the Cross-Layer to CBA Path Manager.

Network Awareness: The status of the network is obtained from the Radio Interface Layer (RIL). From RIL, the CBA Path Manager fetches the required information such as the interface connected (Wi-Fi or Cellular), the cellular network technology behind (3G, 4G, 5G), the radio conditions (SINR). In [17], author research explains that SINR is a good indicator of network quality, so we use SINR as the network awareness parameter. As cross-layer approaches are considered quite complex and resource-intensive, so to make it simple we took an average of the last two intervals and current SINR. Let NA_t is the network awareness at time t, as shown below

$$NA_t = \frac{NA_{t-2} + NA_{t-1} + SINR_t}{3} \tag{2}$$

NA_t values are shown in Table I, and are derived from experiment done by authors in cite [16].

Connection Awareness: The CBA Path Manager fetches connection-specific parameters from the native QUIC library. This includes information such as the end-to-end latency, packet loss rate, receive window, re-transmission rate and other transport-level parameters. We considered the RTT and packet loss rate as connection-specific parameters as most of the literature compared the QUIC performance in these two different values.

The RTT estimation can be done based on the current RTT RTT_t and subsequent previous RTT's. The Smoothed RTT $SRTT_t$ at time t can be estimated as

$$SRTT_t = \gamma \times SRTT_{t-1} + (1-\gamma) \times RTT_t \tag{3}$$

where γ is a constant, from [18] the value of $\gamma = \frac{7}{8}$.

The packet loss rate is taken from the QUIC library. The packet loss rate PL_t at time t is

$$PL_t \leftarrow getCurrentPacketloss()$$

The different value of $SRTT_t$ and PL_t is shown in Table I assigned with different score.

TABLE I
Network And Connection Parameter Data Set

Score	NA_t	$SRTT_t$	PL_t
1	>= 20	0 - 10	<= 10^{-6}
2	13 to 20	10 - 100	10^{-3} - 10^{-6}
3	0 to 13	100 - 300	10^{-1} - 10^{-3}
5	<= 0	>300	>10^{-1}

Context Awareness: The CBA Path Manager fetches the information of application status (foreground or background) from the Android's Activity Manager. The foreground application is active and visible to the user. For example, the YouTube application that renders fetched video from the network. While the background application is not visible to the user but consumes device resources. For example, the user has selected a file to download from Google Drive but currently using some other application. But in the background, a Google Drive file is getting downloaded. To optimize resource utilization by the background application, only one pipe is created. The foreground application's object request is prioritized based on Table II. The above-fetched priority is feedback as input to the CBA-MP-QUIC Classifier.

TABLE II
Object priority for Foreground application

Object Type	Priority
video, audio	highest (4)
javascript	high (3)
html	medium (2)
photo and others	low (1)

B. CBA Classifier

From the information fetched by the CBA Path Manager, the CBA Classifier classifies the network characteristics and connection characteristics as mentioned in Table I. Based on the estimated value, the CBA Classifier assigns three scores as Excellent, Good and Poor. This information is passed to the CBA Session Creator to identify the maximum MultiPipe to be created. It also checks the application characteristics like foreground and background.

C. CBA Session Creator

The creation of multiple pipes impacts the experience, mainly for the short live connections. We have designed the On-Demand Method (OD) and Enhanced On-Demand Method (OD+) methods for the creation of the MultiPipe.

The MultiPipe can be created using the OD method. In this method, if the previously created pipes do not have enough room to fit the new object, a new pipe will be created. This approach is reactive and will increase the latency as it has to establish a new session (pipe creation) after the object arrival. Different from the OD method, we propose a novel OD+ method that pro-actively creates an additional pipe as shown in Fig. 4 Stage 1 and uses this as a backup. This avoids the overhead of creating the pipe after the object's request, henceforth reducing the latency as shown in Fig. 4 Stage 3. We trade-off an additional session creation with latency improvement. OD+ checks the application history of request patterns, like the number of requests from an application when it is in the foreground.

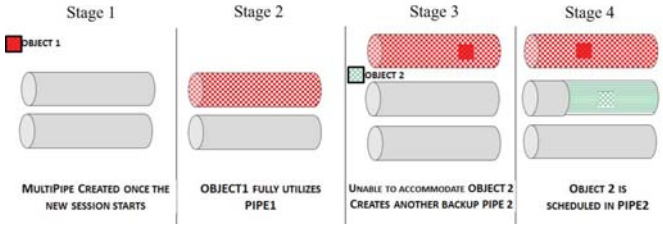

Fig. 4. Enhanced On-Demand MultiPipe Creation in Client

Our CBA Session Creator is modular and can be set to OD or OD+ as per the requirements. In our experiments, the Session Creator uses the OD+ method for the evaluation of MP-QUIC approaches.

The CBA Session Creator receives feedback from the Cross layers and schedules/manages the MultiPipe QUIC efficiently as shown in Fig. 5. The packet processing in each pipe depends on the dynamically varying load. The classified packets are processed in multiple queues using the OD+ method. Our scheduling algorithm guarantees maximum performance with minimal resource utilization. The main reason for the enhancement is the full utilization of the network and adapting to the network conditions. Based on the estimated packet arrival rate and the available burst in each pipe, the packets are scheduled in the Best-Effort MultiPipe, which is explained in the next section.

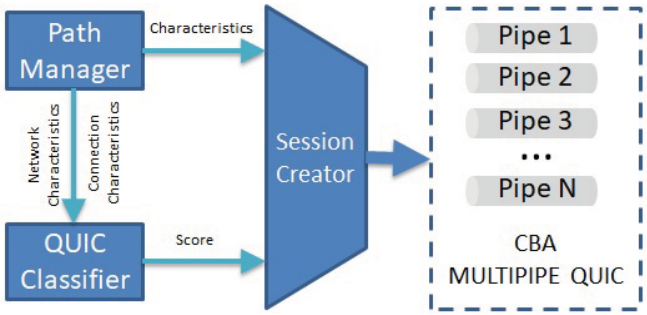

Fig. 5. MultiPipe Creation using the Cross Layer Feedback

V. CBA-MP-QUIC Algorithm

This section explains proposed algorithm. We first provide the MultiPipe creation and initialization followed by the Burst Aware estimation algorithm.

A. Maximum MultiPipe Estimation

The information of network and connection characteristics passed through CBA-MP-QUIC Classifier and the maximum number of pipes for a session are determined. Determining an appropriate number of Maximum MultiPipe to be created, plays a vital role. From our datasets, we classify the conditions into three types as Excellent, Good and Poor with the maximum number of pipes as 1, 3 and 5 respectively. This means, when the current operating conditions are idle, a single pipe would be enough to serve all the objects within a session and as the condition degrades, we increase the number of pipes for better link utilization which decreases PLT. After determining P_j, the CBA-MP-QUIC Session Creator module creates the MultiPipe as shown in Fig. 4.

B. Available Burst Estimation

After determining the maximum number of MultiPipes to be created and the Per-Pipe Buffer size P_j, we derive the available burst estimation of each pipe. This is a dynamic module, which updates based on object download start/progress/completion.

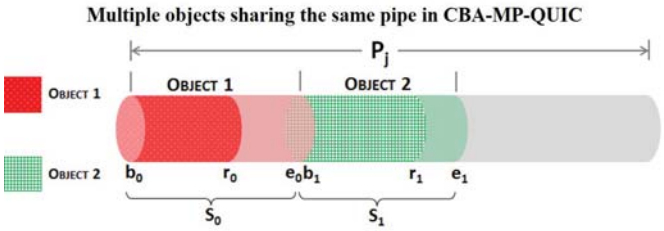

Fig. 6. Burst Available Estimation

In the above Fig. 6, multiple requests are using the same pipe, so to we compute the bust availability in following way:
$b_i \leftarrow$ Beginning of object i
$r_i \leftarrow$ Received of object i
$e_i \leftarrow$ End of object i
$S_i \leftarrow$ Total Size of object i
Hence, the received size of object i, R_i can be written as

$$R_i \leftarrow r_i - b_i \tag{4}$$

Hence, from (4) total available burst for pipe j is the difference between Pipe Buffer Size and sum of all object received size. It can be derived as:

$$BA_j \leftarrow P_j - \sum_{i=1}^{n} R_i \tag{5}$$

Where n is number of object request for a pipe

C. CBA MultiPipe Management

We consider the packets arrive at an exponentially distributed inter-arrival time, i.e Poisson arrival [20] at a sub-second interval. Usually, the TCP traffic follows Poisson [21] and hence we model QUIC [2] as Poisson arrival considering the same. The service rate of the CBA Session Creator is μ bits per sec. The packets arriving at the scheduler are serviced on a class-based FIFO (first in first out) queuing discipline for a fixed quantum. For a pipe j the service rate μ depends on the available burst BA_j. There would be multiple numbers of applications (APPs) running and would have opened multiple sockets. For a packet in a socket i the mean arrival rate of the packet is given by λ_i. Therefore, the total mean arrival rate λ at the scheduler can be written as $\lambda = \sum_{i=1}^{n} \lambda_i$, where n in the total number of object request in a pipe.

For best effort utilization, the total mean arrival rate must be always less than the service rate of the scheduler i.e. $\lambda \leq \mu$. The service rate of the socket i can be written as $\alpha_i = \frac{\lambda_i}{\lambda}\mu$. The service times is a deterministic time D serving at rate $\mu = \frac{1}{D}$. The service duration of packets in socket i with length L is $D_i = \frac{L}{\alpha_i}$, which is of fixed duration. Hence, we use $M/D/1$ queue model for scheduler where

- Arrival rate $\rightarrow \lambda$
- Service rate $\rightarrow \mu$
- Utilization $\rightarrow \rho = \frac{\lambda}{\mu}$

For the packet in socket i, the average waiting time in the queue $E[W_i]$ is given below as derived in [22]

$$E[W_i] = \frac{1}{2\mu}\left(\frac{\rho}{1-\rho}\right) \tag{6}$$

L_q is the average length of the queue required at any point of the time, so that packets are not lost. The L_q for the system can also be obtained using the Little's Law [23].

$$L_q = \lambda E[W_i] \tag{7}$$

The packet loss happens when the particular queue overflows i.e. $\rho > 1$. This means that the packets are arriving faster than they can be serviced. When the BA_j at a point is equal to L_q, then packet losses can occur. To avoid the packet losses when BA_j and L_q equal, due to packet overflow we used the β factor whose value is 1.2. The difference between the burst available and required queue length can be written as below

$$\alpha = BA_j - \beta L_q \tag{8}$$

$$\alpha = \begin{cases} > 0, & \text{Pipe can be used for new request} \\ <= 0, & \text{Pipe is full, control to Session Creator} \end{cases}$$

The BA_J is the available burst at a particular time t for pipe j, which is calculated dynamically. When the average length of the queue (L_q) at the same time t is less than BA_j then the pipe j can be used for new requests, otherwise the j pipe is full and Session Creator is updated about burst.

D. CBA-MP-QUIC Algorithm

The backbone of CBA-MP-QUIC is to select the optimal pipe amongst all MultiPipe. The design of Algorithm 1 guarantees the following:

- The high priority object is allocated to the pipe with the maximum available burst
- The pipe is created if and only if it is required
- Minimal number of pipes has to be used at any point of time

Algorithm 1 CBA-MP-QUIC Algorithm

INPUT: $P_{created\_count}$ {Current count of pipes created}
$P^{P_{created\_count}}$ {Pipes created till now}
OUTPUT: P_{result}
$conn_{cellular} \leftarrow celluarConnected()$ {3G, 4G, 5G}
$NA \leftarrow score(conn_{cellular}, NA_t)$ {NA_t from (2)}
$CA \leftarrow \max(score(SRTT_t), score(PL_t))$ {from (3)}
$Pipe_{max} \leftarrow min(NA, CA)$
$appStatus \leftarrow getAppStatus()$
if $appStatus.isForeground$ = true **then**
  $OBJ_{priority} \leftarrow getReqObjectPriority()$
  $\alpha^{P_{created\_count}}$ {array to store values from (8)}
  $isBurstAvailable \leftarrow false$
  $j = 0$
  while $j < P_{created\_count}$ **do**
    $\alpha^j = get\alpha ForPipe(P^j)$ {from (8)}
    if $\alpha^j > 0$ **then**
      $isBurstAvailable \leftarrow true$
    end if
  end while
  if $isBurstAvailable = true$ **then**
    if $OBJ_{priority} > high$ **then**
      $resultPipe \leftarrow getBest\alpha AvailablePipe(\alpha^{1...P_{created\_count}})$
    else
      $resultPipe \leftarrow getBestFit\alpha AvailablePipe(\alpha^{1...P_{created\_count}})$
    end if
  else
    if $Pipe_{max} > P_{created\_count}$ **then**
      $P_{created\_count} \leftarrow P_{created\_count} + 1$
      $P^{P_{created\_count}} \leftarrow createNewPipe()$
      $P_{result} = P^{P_{created\_count}}$
    else
      $max\_\alpha_{index} = max\alpha\_index(\alpha^{1...P_{created\_count}})$
      $P_{result} \leftarrow P^{max\_\alpha_{index}}$
    end if
  end if
else
  $P_{result} \leftarrow P^0$ {Background app will always have 1 pipe}
end if
return P_{result}

The above Algorithm 1, $P_{created\_count}$ is the number of pipes created till now. $P^{P_{created\_count}}$ are the created pipes. With the cellular network technology, the network and connection parameters score is decided. The score is given as input to decide the maximum pipes $Pipe_{max}$ that can be created. Based on the application status, object priority, and available burst of each pipe is compared to provide the best available pipe to the object request.

VI. Result and Discussion

This section explains the experimental setup used for evaluation followed by an in-depth analysis of results. The experiment is carried out in the varying network (SINR, RTT) and operational conditions (server location and web page content).

A. Experimental Setup

For the testbed, the Chromium code [24] was forked from Google Chromium git (V 66.0.3358.0) for the experiment. At the webserver end, the QUIC_server runs on Ubuntu 18.04.3 LTS supporting Linux kernel 4.20. On the client end, Samsung S8 and S10 variant with Chromium Browser having QUIC v39 running on top of Android Oreo was used to download the content from the QUIC server as shown in Fig. 7. The web page parameters as shown in Table III, Page type (small, medium and large), the total size of each page, objects from minimum to maximum size, total page size and number of objects present in a page.

TABLE III
Server hosted with different page size setup

Page	Min-Max object Size	Page Size	#Object
Small	7.3KB - 197KB	580 KB	8
Medium	7.3KB - 1.7MB	3.7 MB	13
Large	7.3KB - 5.7MB	16.9 MB	22

For live air testing, the servers were hosted at Samsung Labs of Suwon, South Korea, and Bangalore, India, as shown in Fig. 7. The cellular network of 4G/3G was used to connect the QUIC servers at Samsung labs India and Korea. Different HTML pages were fetched as mentioned in Table III in parallel at the Samsung S10 and S8 devices.

B. Live Air Experiment of MP-QUIC Solutions

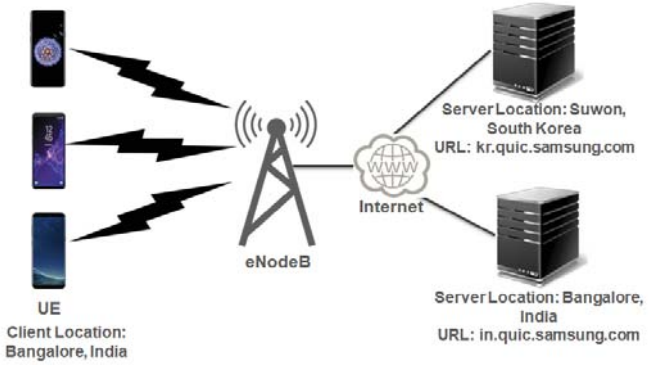

Fig. 7. LiveAir Setup with Cellular Network

- RR-MP-QUIC: The PLT of RR-MP-QUIC and legacy QUIC is compared with the different HTML pages that were fetched as mentioned in Table III from different servers located in Samsung Lab India and Samsung Lab Korea. The pages were fetched multiple times on different devices to average out PLT. The performance of RR-MP-QUIC as shown in Fig. 8 is always better than the legacy QUIC. RR-MP-QUIC performed best when the large page is fetched as the number of objects and page size is more. The performance of RR-MP-QUIC is 76% better than legacy QUIC because the RR-MP-QUIC better utilizes the network bandwidth with MultiPipe as mentioned in Section III. With an increase in network load and latency, the bandwidth utilization of RR-MP-QUIC is better as compared to legacy QUIC.

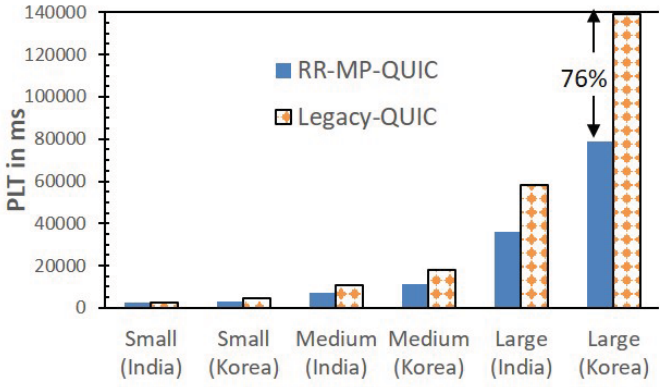

Fig. 8. Live Air PLT comparison of RR-MP-QUIC vs Legacy QUIC

- CBA-MP-QUIC: The experiment setup is the same as that of RR-MP-QUIC. Even the trend as shown in Fig. 9 is the same that of RR-MP-QUIC but the performance is better because the CBA-MP-QUIC creates the MultiPipe when required and intelligently assigns the best possible burst available pipe to object request as mentioned Section IV.

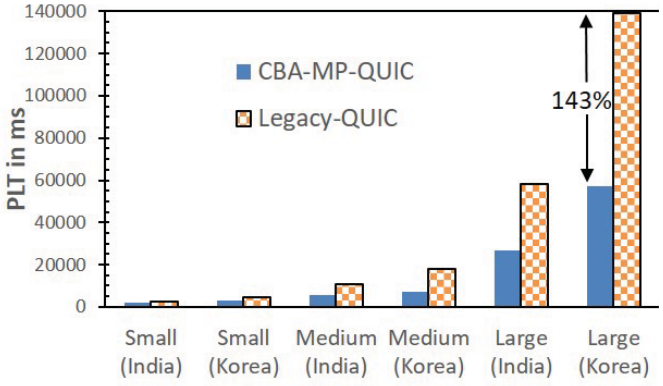

Fig. 9. Live Air PLT comparison of CBA-MP-QUIC vs Legacy QUIC

The performance of CBA-MP-QUIC in the live air experiment is better when the same content is fetched from the Samsung Korea server as the RTT is more and losses to the end medium to UE are more. Thus CBA-MP-QUIC performs best in the current solution in different network conditions.

C. Fairness Considerations

Apart from PLT and throughput, fairness is one of the important metrics for evaluation. Mainly when multiple connections are opened for the same page, fairness is an important criterion to be looked upon. This is because opening multiple pipes or connections might reduce the PLT in a device but might impact the performance of other devices connected to the same server or the same network. In this section, we evaluate the fairness of CBA-MP-QUIC. In our experiments, three devices of the same variants fetch the same content from the server at an interval of 5 seconds. The bandwidth of the channel was tuned to 5Mbps with no simulated packet loss.

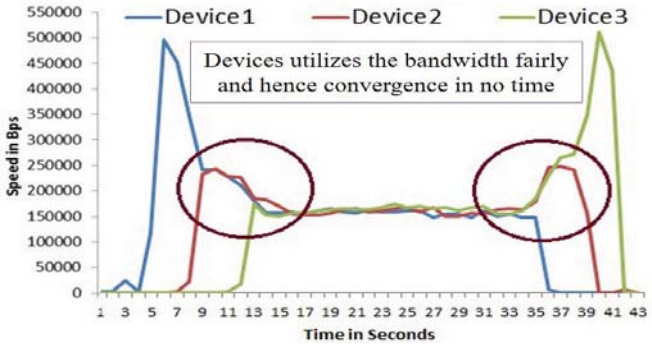

Fig. 10. Intra Fairness Evaluation of CBA-MP-QUIC

Device1 is first to requests the content and after 5 seconds from the time Device1 starts, Device2 begins to requests the content. Similarly, Device3 requests the contest after 5 seconds from the time Device2 starts. Fig. 10 depicts the server-side bandwidth shared among the devices. From the graph intra-fairness of CBA-MP-QUIC from which we can observe that the bandwidth sharing is uniform. In CBA-MP-QUIC, due to the information from the cross layers, optimally creates the pipe when required as mentioned in Section IV. CBA-MP-QUIC reaches the convergence point where the devices fairly share the network resources.

VII. Conclusion

Through this paper, we investigated and addressed the underutilization problem of QUIC in wireless mobile networks. We proposed two novel protocols for MultiPipe QUIC (MP-QUIC) protocol. The proposed RR-MP-QUIC protocol, which schedules the objects in a round-robin manner, decreased the web page loading time by 76%. Furthermore, to improve the RR-MP-QUIC performance, a novel Cross-Layer Burst Aware (CBA) MP-QUIC was proposed, which efficiently manages the MultiPipe understanding the Cross-Layer wireless characteristics. From the live air experiments, it is observed that the CBA-MP-QUIC solution outperforms legacy QUIC by up-to 143%. The convergence point of bandwidth among the devices was efficient, proving it to be a fair and efficient algorithm.

Our future work presupposes a cross-layer bandwidth detection from the modem and dynamic control of MultiPipe to increase the accuracy of Bandwidth and handover estimations.

References

[1] "TCP slows down HTTP/2", [online] Available: https://community.akamai.com/customers/s/article/How-does-HTTP-2-solve-the-Head-of-Line-blocking-HOL-issue

[2] "QUIC: A UDP-Based Multiplexed and Secure Transport", [online] Available: https://datatracker.ietf.org/doc/draft-ietf-quic-transport

[3] D. Murray, T. Koziniec, S. Zander, M. Dixon, and P. Koutsakis, "An Analysis of Changing Enterprise Network Traffic Characteristics", Proc. of IEEE APCC, 2017

[4] N. Oda, and S. Yamaguchi, "HTTP/2 Performance Evaluation with Latency and Packet Losses", Proc. 15th IEEE Annual Consumer Communications & Networking Conference (CCNC), 2018

[5] P. Qian, N. Wang, and R. Tafazolli, "Achieving Robust Mobile Web Content Delivery Performance Based on Multiple Coordinated QUIC Connections", IEEE Access, Vol 6, 2018

[6] Choudhary, Gunjan Kumar, et al. "Method and system for handling data path creation in wireless network system." U.S. Patent Application No. 16/384,040.

[7] G. Carlucci, Luca De Cicco, and Saverio Mascolo,"HTTP over UDP: an experimental investigation of QUIC", Proc. ACM Symposium on Applied Computing, 2015

[8] M. Ansari, and M. Ghaderi, "Parallel HTTP for Video Streaming in Wireless Networks", IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2016

[9] Arunachalam, Karthikeyan, et al. "Layer 4 accelerator (l4a) for optimizing network protocol latencies in mobile devices." 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). IEEE, 2018

[10] Sabareesh, Dronamraju Siva, et al. "Redundant TCP Connector (RTC) for Improving the Performance of Mobile Devices." 2019 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2019

[11] Ppallan, Jamsheed Manja, et al. "Flare-dns resolver (fdr) for optimizing dns lookup overhead in mobile devices." 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2019

[12] D. Bhat, A. Rizk, and M. Zink, "Not so QUIC: A Performance Study of DASH over QUIC", Proc. 27th Workshop on Network and Operating Systems Support for Digital Audio and Video, 2017

[13] S. Cook, B. Mathieu, P. Truong, I. Hamchaoui, "QUIC: Better for what and for whom?", Proc. ICC pp. 1-6 May. 2017

[14] Kanagarathinam, Madhan Raj, et al. "D-TCP: Dynamic TCP congestion control algorithm for next generation mobile networks." 2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2018

[15] K. Nepomuceno, I. Oliveira, R. Aschoff, D. Bezerra, M. Ito, W. Melo, D. Sadok, and G. Szabó, "QUIC and TCP: A Performance Evaluation", Proc. IEEE ISCC, 2018

[16] J. Parikh, and A. Basu, "Effect of mobility on SINR in long term evolution systems", Proc. Commun. Technol., vol. 7, no. 1, pp. 1239-1244, Mar 2016

[17] G. Basilashvili, "Study of Spectral Efficiency for LTE Network", Proc. ASRJETS, Volume 29, No 1, pp 21-32, 2017

[18] "QUIC Loss Detection and Congestion Control", [online] Available: https://tools.ietf.org/html/draft-ietf-quic-recovery

[19] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental evaluation of multipath TCP schedulers." Proc. of ACM CSWS, 2014.

[20] T. Karagiannis, M. Molle, M. Faloutsos, and A. Broido, "A nonstationary poisson view of internet traffic", Proc. IEEE INFOCOM, pp. 1558-1569, Mar. 2004.

[21] A. Chydzinski, D. Samocuik, B. Adamczyk, "Burst ratio in the finite-buffer queue with batch Poisson arrivals", Applied Mathematics and Computation, Aug. 2018, vol. 330, p. 225–238.

[22] C. Robert S, "Wide Area Network Design:Concepts and Tools for Optimization", 1998, p. 319. ISBN 1558604588.

[23] Little, J. D. C.; Graves, S. C. (2008). "Little's Law". International Series in Operations Research & Management Science. 115. p. 81. doi:10.1007/978-0-387-73699-0_5. ISBN 978-0-387-73698-3.

[24] "QUIC A Multiplexed Stream Transport Over UDP", Dec. 2017 [online] Available: https://www.chromium.org/quic.