On Ultra-Reliable Low Latency Multipath TCP For Connected Autonomous Vehicles

Shiva Raj Pokhrel, Neeraj Kumar and Anwar Walid

Abstract—Connected Autonomous Vehicles (CAVs) are Not-So-Futuristic. CAVs will be highly dynamic by intelligently exploiting multipath communication over several radio technologies, such as high-speed WiFi and 5G and beyond networks. Yet, the likelihood of data communication loss can be very high and/or packets arrive at the destination not in correct working order due to erratic and mixed time-varying wireless links. Furthermore, the vehicular data traffic is susceptible to loss and delay variation, which recommends the need to investigate new multipath TCP (MPTCP) protocols for ultra-reliable low latency communication (URLLC) over such heterogeneous networks while reassuring CAVs' needs. We undertake the challenge by jointly considering network coding and balanced linked adaptation for performing coupled congestion control across multiple wireless paths. Consequently, the proposed low delay MPTCP framework for connecting autonomous vehicles is efficient and intelligent by design. We conduct a rigorous convergence analysis of the MPTCP design framework. In summation, we provide a detailed mathematical study and demonstrate that the latency penalty for the URLLC-MPTCP developed over these networks becomes negligible when considering the possible benefits that multiple network convergence could offer. Our extensive emulation results demonstrate all these lucrative features of URLLC-MPTCP (see [1]).

Index Terms—Multipath TCP, Streaming Network Coding, Error Correction, Balanced Load Adaptation, Connected Autonomous Vehicles.

I. BACKGROUND

THE global autonomous driving market is forecast to be worth US\$173.15 billion by 2030¹ and we are committed to leveraging these technologies to improve the safety, efficiency and sustainability of its transport system. Connected Autonomous Vehicles (CAVs) are important for several applications ranging from robust communication for driving a platoon of military vehicles together [2], and broadcasting safety messages to/from emergency vehicles on the move, to the entertainment services on the way. Cars shall certainly talk and discover other vehicles either directly or via roadside units to prevent collisions at the intersection or for spreading notifications about critical situations and bicycles or pedestrians on the path. The need to simultaneously utilize heterogeneous wireless networks, both macro and micro cells of 5G along with the existing wireless infrastructure build on the largely

Fig. 1. CAVs are communicating with a Cloud Server on the Internet via 5G and WiFi links. CAVs also form an adhoc network using WiFi like dedicated short-range communications (DSRC). Each of the wireless links uses its own code rate R_i to combat lossy links. Links have their own packet loss characteristics (say on an average with loss probability e_i) and effective latency of d_i seconds per packet. As a result, the perceived information rate via link i is R_i/d_i packets per seconds.

deployed WiFi and emerging 5G/6G standards, has already created an ideal scenario for the CAVs [3]–[5].

CAVs will appear very soon by simultaneously using 5G, high-speed WiFi and multiple other connections. A *multipath* transmission control protocol (MPTCP) has already been standardized as a tool for harnessing bandwidth by IETF [6]. It shed light on the possibility of i) aggregating multiple radio technologies (WiFi/5G) for enhancing CAVs' performance, ii) allowing improved and seamless handoffs, and iii) facilitating robust connectivity at all times [7], as illustrated in Fig. 1. For example, with MPTCP, one can theoretically simplify driver experience by improved network loss tolerance as well as better network performance. Clearly, MPTCP facilitates some major advantages for reliable and stable vehicular connections, but it [8] has a range of challenges to be resolved in order to realize its practicability over CAVs.

Although integration of existing wireless standards (WiFi/3G/4G network convergence) and 5G has the potential to dramatically improve efficiency for CAVs' data communications, frequent packet failures due to channel losses and mobility, highly changing topology of networks, frequent handovers, unstable connections, etc. have serious implications for meeting CAVs'—*ultra reliable low latency communication* (URLLC)—requirements in real time [9], [10]. A crucial issue of MPTCP concerns *data* transmissions and their responses (*acknowledgments*) for stable communication, in particular for connectivity along with multiple links, including such wireless routes, which often have separately differing link quality and sporadic delays.

S. R. Pokhrel is with the School of Information Technology at Deakin University, Geelong, Australia. N. Kumar is with the Department of Computer Science and Information Engineering, Asia University Taiwan. A. Walid is with Nokia Bell Labs, Murray Hill, NJ, USA. The revised version of this work is accepted for publication in IEEE TVT [1] Email: shiva.pokhrel@deakin.edu.au

¹Frost and Sullivan, Global Autonomous Driving Market Outlook, March 2018.

When available, CAV could use multiple network interfaces and wireless links dynamically with MPTCP. Experimental research works have already shown that, for bulk transfers of files over links of similar quality, MPTCP allows remarkable gains over traditional TCP and that data transport during handoffs also occurs fairly smoothly [11]–[13]. Nevertheless, MPTCP has been inheriting the following major drawbacks from single-path TCP [4], [14], [15]:

1. *Retransmission of lost packets*: A missing packet must be identified and re-transmitted by the sender (which requires long delays, RTTs involved in loss discovery and selective repeat process).
2. *Need for in-order delivery of packets*: Reordering of packets at the receiver end severely affects the efficiency of MPTCP where links are of a significantly different nature (e.g., WiFi vs 5G) [13], [16], [17].

If the link carrying the *data* and/or *ack* packets fails due to a fading in the wireless channel, the transmission and load balancing mechanism of MPTCP data transport will be largely disrupted. Furthermore, if a packet transmission fails on one of the routes, another (good) link will get starved, resulting in CAVs output being compromised and deteriorating drivers' experience. But, emergency and safety-critical CAVs data traffic are susceptible to such high and oscillating latencies. Also, the likelihood that packets will experience failure or reach unordered at the destination via heterogeneous, time-varying and ergodic wireless links, is quite high. Therefore, all missing and unordered packets must be retransmitted or buffered at the receiver for resequencing before they can be sent in sequence to the CAV applications.

In contrast to our earlier works [4], [15], in this paper, we study and exploit to ameliorate the aforementioned two impending issues of MPTCP over CAVs as follows. Specifically, the ultra-reliable low latency communication URLLC requirements [18] for stable CAVs traffic in such heterogeneous (convergent) networks could be minimized significantly as follows:

1. reducing the resequencing latency with *load-balanced link adaptation* across multiple links; and
2. using *streaming code* that performs encoding to counter packet losses and reduces not only the retransmission but also resequencing latency of packets [4], [19], [20].

By jointly exploiting *load balanced link adaptation* and *Stream Coding*, it is plausible to optimally assign coded and information packets over the respective links timely [8], with *guaranteeable delay*. Load balanced link adaptation aims to optimize network resource use, maximize throughput, minimize response time, and avoid overload of any single link.²

Overall, in this work, we design to mitigate the adverse impact of erratic wireless losses and unordered delivery to attain guaranteeable delay and high reliability. Retransmissions required for packets over erroneous links are substantially

minimized and/or the latency incurred for ordering packets at the receiver end is completely solved.

A. Novelty and Contributions

Development of reliable and efficient MPTCP for CAVs: We design a new ultra-reliable low latency MPTCP for the CAVs, refer to as 'URLLC-MPTCP', that performs coding and transfers packets to dynamically changing multiple wireless network links in a way that manages the resource utilization and increases the performance (throughput). With this new approach MPTCP seldom goes for the undesirable retransmit, timeouts and recovery phases at the transport layer.

Two main contributions are to this end are:

1. We evaluate the URLLC-MPTCP protocol extensively for the validity of the presumptions made during its development.
2. We develop the convergence and consistency conditions of our URLLC-MPTCP protocol to investigate its impact over the network dynamics.

Mathematical modelling of URLLC-MPTCP: We develop a rigorous mathematical analysis for the URLLC-MPTCP protocol to analyze the packet re-sequencing delay at the receiver due to path heterogeneity [4].

Of particular relevance to this work are the pioneering works on multipath transport in [22]–[24], which establish the foundation for the development of MPTCP *linked-increases algorithm* (LIA) [6]. However, Khalili *et al.* [25] demonstrated that LIA [6] is not *Pareto-optimal*³ and developed an *opportunistic linked-increases algorithm* (OLIA) [25]. Moving ahead, Peng *et al.* [26] presented comprehensive analysis for designing MPTCP and proposed *balanced linked adaptation* (BALIA) [27].

B. MPTCP over CAVs: Problems

To completely leverage the capacity of bulk MPTCP connections over CAVs, we have discovered the following key challenges to be tackled. It is important to understand in this regard that the 3G/5G and WiFi networks have completely different channel dynamics for data transmission. In a 3G/5G network, each MPTCP connection is assigned specifically to a different (dedicated) channel and has a wide geographical coverage. In contrast, the coverage is not only limited to WiFi networks, but also all (shared) connections must use the same wireless platform. Such sharing and small coverage in WiFi always contributes to repeated handoffs, over-access contention and packet collisions for the CAVs. On the other hand, the (cellular) route in 3G/5G networks requires typically large buffering memory; consequently, in long-oscillating latencies in combination with small loss rates, whereas the WiFi link corresponds to short-steady latencies but a relatively higher rate of packet loss [28].

There is a highly ergodic and time-variable probability of packet loss and delays while transporting data over such wireless links. It is influenced by several network variables,

²Stream code is based on *fountain or rateless codes* [21, pp 589-594] and also due to the potential benefits that these codes deliver, for example, they do not exhibit a fixed *code rate*. It is known in information and coding theory [21] that the code rate is a *proportion of the data stream that is non-redundant* (useful).

³A condition in which it is improbable to obtain anyone better off without performing someone worse off is called Pareto optimal condition.

such as handoffs, resource buffering memory sizes, queuing policy, other connections and users competing in the links, wireless errors, retry limits, etc. Hence, as packets are sent through different WiFi and cellular routes, they may easily be lost or reach unordered at the destination. Therefore, the misplaced (lost) packets must be retry (resequencing) until the CAVs' applications receive them. This requires a large recipient buffering memory and corresponding high retransmissions at both link and TCP layers, thus undermining the performance of MPTCP.

II. DESIGN THEORY OF URLLC-MPTCP

Our MPTCP design idea consists of the following three main advancements in knowledge.

1. **STREAMING CODE.** We exploit streaming code to combat link errors and reordering delays; as a result, MPTCP senders with increased redundancy in packet transmission could quickly suck up erratic delays and losses over the wireless links [4], [20].

2. **BALANCED LINK ADAPTATION.** In contrast to seminal MPTCP design approaches, we adopt a balanced link adaptation in our approach, incorporating instantaneous throughputs, losses and delays experienced by the preceding packets on the links. URLLC-MPTCP is capable of changing its action in response to varying link situations and past experience. For an efficient and experience-driven MPTCP design, we need to predict the characteristics of the wireless links, but existing MPTCP congestion control protocols [6], [8], [25], [26] do not account for the instantaneous network variation and wireless impairments across multiple paths. As a result, earlier works [6], [8], [25], [26] are not capable of exploiting time-varying topology and link diversity of the CAVs networking.

With the aim of tackling delay variation in wireless networks, each MPTCP data packet from the sender records its originating timestamp, which is then used by the MPTCP sink upon reception of the packet to compute *one-way delay*. The computed value of the *one-way delay* is then feededback to the sender, piggybacked in the acknowledgment packet. This new idea develops a mechanism for the proposed URLLC-MPTCP to simultaneously exploit and respond to dynamic topology, channel impairments, and link diversity of the CAVs networking.

3. **GUARANTEEABLE DELAY.** With relevant insights from [29], we adopt the notion of guaranteeable delay ($D_{3\sigma}$), which is the expected mean delay ($\mathbb{E}[D]$) plus three standard deviations ($3 \times \sigma$) of the mean. This is motivated by the celebrated *three-sigma rule of thumb*, which expresses a conventional heuristic that nearly all delay values are taken to lie within three standard deviations of the mean, and thus it is empirically useful to treat 99.7% probability (as near certainty) that

$$Pr(\mathbb{E}[D] - 3\sigma < D < \mathbb{E}[D] + 3\sigma) \approx 99.73.$$

Therefore

$$\mathbb{E}[D_{3\sigma}] \approx \mathbb{E}[D] + 3\sigma.$$

This novel idea develops a mechanism for the proposed URLLC-MPTCP to exploit and respond to dynamic topology,

channel impairments, and link diversity of the CAVs networking simultaneously with guaranteed permissible delay.

A. Proposed URLLC-MPTCP

Each URLLC-MPTCP sender takes latency D (in seconds) as a delay constraint from the application layer and has a set of links. Every single link manages a separate congestion window ω_i and measures its (running averages) instantaneous one-way delay and round trip time, τ_i seconds. Let $\hat{d}_i$ be the running average delay in seconds per packet from the sender to the road user on link i . Then the window adaptation algorithm is designed as follows.

Algorithm 1 Proposed URLLC-MPTCP protocol

Output: $\omega_i, \hat{d}_i, \tau_i$

```

1: while (Congestion Avoidance) do
2:   If (packet loss signal on link  $i$ )
3:     Update  $\omega_i$ :  $\omega_i \leftarrow \omega_i/2$ 
4:     Transmit  $\lceil \omega_i/R_i \rceil$  coded packets
5:   Else
6:     Update  $\omega_i$ :  $\omega_i \leftarrow \omega_i + \min\{1/\omega_i, \alpha(\omega_i)\}$ 
7:     Transmit  $\lceil \omega_i/R_i \rceil$  coded packets
8:     Update  $\hat{d}_i$ :  $d_i \leftarrow \text{average}\{d_i\}$ 
9:   end while
```

As shown in Algorithm 1, for every effective packet loss indication over link i , we perform multiplicative decrease and update the congestion window ω_i as $\omega_i \leftarrow \omega_i/2$. Thereafter, to combat with channel errors opportunistically, we perform coding with rate R_i and send $\lceil \omega_i/R_i \rceil$ coded data packets on the link i . In contrast to the multiplicative decrease for a packet loss, we perform continuous increase in the congestion window ω_i for every successfully transmitted packets over link i as $\omega_i \leftarrow \omega_i + \min\{1/\omega_i, \alpha(\omega_i)\}$. Then, we perform coding to generate $\lceil \omega_i/R_i \rceil$ coded packets, with parameter

$$\alpha(\omega_i) = \frac{1}{\beta_i} \max_j \left\{ \frac{\omega_j}{\tau_j^2} \right\} / \left(\sum_j \frac{\omega_j}{\tau_j} \right)^2. \quad (1)$$

where β_i is the load balancer defined as

$$\frac{1}{\beta_i} = \min\{D_{3\sigma}, d_i\} \times \frac{\text{Number of links}}{\sum_i d_i}. \quad (2)$$

Complexity of Algorithm 1: Consider a situation when $\alpha(\omega_i) > 1/\omega_i$, then the window update Eqn. (1) coincides with that of conventional single-path TCP. It is worth noting that the continuous increment phase is not strictly linear as in seminal TCP and MPTCP protocol; rather, it is adaptive and performs load balancing in flight by observing other sub-flows. Furthermore, when $\hat{d}_i$ is the same along with all links, then (β_i becomes 1), as a result, the window update Eqn. (1) coincides with MPTCP LIA enhanced with stream code (i.e. *LIA plus coding*). Our adaptive load balancing idea with an inclusion of a *load balancer*, β_i , (see Eqn. (2)) during window evolution, tackles the transient bandwidth variation over CAVs network in real-time and guarantee low latency (min operator in Equation (2) guarantee $\hat{d}_i < D_{3\sigma}$) for satisfying latency requirements of higher layer real-time applications. The complexity of finding

the minimum in step 6 (Algorithm 1) is linear in the number of paths (see Appendix for details).

B. Streaming Code Policy

Motivated by the analyses in [4], [20], we consider a CAV link i has a likelihood of packet loss e_i and delay d_i (seconds per packet). With a code rate R_i for the link, we have the following inequality that the code rate, R_i , satisfies for all links

$$\sum_i \frac{(1 - e_i)^2}{d_i} > \sum_i \frac{(1 - e_i)R_i}{d_i} \quad \text{i.e.} \quad \sum_i \frac{1}{d_i} (1 - e_i)((1 - e_i) - R_i) > 0. \quad (3)$$

Observe in Eqn. (3) that the code rate must satisfy ($0 < R_i < 1$) and $(1 - e_i) > R_i$, (as e_i can be perceived a form of *packet erasure probability* [21]).

III. DELAY MODELING OF URLLC-MPTCP

We have the following two assumptions for pragmatic modelling.

ASSUMP1: *Coded packets are transmitted only over one link, say i , thus the streaming code rate R_i always satisfies the following condition:*

$$R_i < 1 - \frac{\sum_{j \neq i} (1 - e_j)e_j/d_j}{(1 - e_i)/d_i}. \quad (4)$$

Eqn. (4) is motivated by the analysis reported in [4], [20]. One condition which assumes a network scenario where URLLC-MPTCP user suffers from severe link fading impairments along with one of the links when all other links are in perfect condition, where it might be easier to transfer coded packets only over a single link (thus effectively utilizing all links) i , which contributes to the Eqn. (4), providing us all with a unique coding probability formula.

When the packets are sent over two separate internet connections with d_i and d_j ($d_i > d_j$) delays. Packets sent over the line j should wait for a further latency of $(d_i - d_j)$ seconds.

ASSUMP2: *Packets sent through the high-speed links are still waiting for the packets arriving from the slower links to be sent to the client application (in a higher layer) in time, for sequential delivery of packets to the application in order.*

ASSUMP2 impacts the study as there is a (very small) probability that packets sent from the high-speed connections will be distributed in order, and that there is no waiting for the packets from the slow connection.

We make use of the following recursive reward theoretical approach. Using the results of *renewal-reward approach*, the delay for in-order delivery of packets by the URLLC-MPTCP can be computed by estimating the expected number of packets sent over the coded link.⁴ Using the T_n lifetime renewal period

⁴Indeed, a recurrent cycle happens when streaming coded packets are received at the end-user as a consequence, in a decoding case. A recursive event occurs (# of coded packets received $\geq$ # of missing packets of information).

and the mutually exclusive U_n bonuses, $n \geq 1$ (U_n will rely on T_n). When we pair a rewarding utility with a renewal time period, in this replacement incentive system, $U(t)$ is the cumulative incentive utility up to t , the rate of reward amount is then given by⁵

$$\lim_{t \rightarrow \infty} \frac{U(t)}{t} = \mathbb{E}[U_2]/\mathbb{E}[T_2]. \quad (5)$$

Let time sequences $T_1, T_2, \dots$, where $T_m = 0, 1, 2, \dots$ time-slots, be the iid (inter)arrival time points between succeeding decoding events with first moment $\mathbb{E}[T]$ and second moment $\mathbb{E}[T^2]$ respectively. The arrival can be viewed as a random variables process (arrival epochs), $0 \leq Y_1 \leq Y_2 \leq Y_3 \leq \dots$ where the m^{th} epoch $Y_m = \sum_i^m T_m$. For tractability of our analysis, we have the following simplifying approximation:

APPROX1: The distribution is Poisson for $z_{n,i} = 0, 1, 2, \dots$

$$\Pi(z_{n,i}) = e^{-\lambda} \frac{\lambda^{z_{n,i}}}{z_{n,i}!}; \quad \lambda = \sum_i \frac{x_i e_i}{x_c} + e_c; \quad (6)$$

where the fraction x_i/x_c denotes the approximate rate of information packets per link per coded packet.

The volume of lost packets in a time slot is given by Eqn. (6) following an iid process. The timings between decoding events and satisfy Eqn. (4) (recall ASSUMP1) for all e_i and d_i of all active links:

$$\pi(t_n) = \lambda^{t_n} e^{-\lambda} \frac{(t_n - 1)^{(t_n - 2)}}{t_n(t_n - 2)!}, \quad t_n > 1. \quad (7)$$

Therefore, along the lines of [4] for one or two links, i.e., $i \in \{1, 2\}$

$$\mathbb{E}[T^2] = \frac{e^{-\lambda} \lambda (1 - \lambda + \lambda^2)}{(1 - \lambda)^3}; \quad \mathbb{E}[T] = \frac{e^{-\lambda} \lambda}{1 - \lambda}.$$

The $\mathbb{E}[T] < \infty$ and $\mathbb{E}[T^2] < \infty$, the packet loss rate over all available links should satisfy $\lambda < 1$. It not only mandates Eqn. (4) but also holds true for all other active links $e_i/(1 - R_i) \approx 1$.

A. Estimating Mean Delay

Using Eqn. (5), the delay can be predicted as

$$\mathbb{E}[D] \approx \lim_{t \rightarrow \infty} \frac{U(t)}{t} = \mathbb{E}[U_2]/\mathbb{E}[T_2]. \quad (8)$$

To approximate the $\mathbb{E}[U_2] < \infty$ and simplify our analysis, we adopt the following approach.

We find a part of the cycle where $T_2 = \max(2, T_n)$, i.e., $T_2 \in 2, 3, \dots$, is the slot between new process arrivals. The compensation feature for evaluating the latency faced by information packets transmitted on the link u is very close to the approximated process's residual existence. Therefore,

$$\mathbb{E}[U_2] \approx \sum_i \sum_{t_n=2}^{\infty} \mathbb{E}[U_2|u=i, T_n=t_n] \pi(t_n) I_u(i);$$

where $I_u(i)$ indicating coded or uncoded link.

⁵The reason for the subscript 2 of Eqn. (5) is as follows. U_n and T_n are i.i.d, and the effect of the initial cycle values is smallish on the limit. We have applied a similar idea in [4].

Fig. 2. Two-state wireless channel modelling.

B. Modelling Guaranteeable Delay

With relevant insights from [8], [29], we consider two-channel states: good state (G) and bad state (B). We develop expressions for the probability distributions of the channel state as follows. Considering discrete time-slots, where the duration of each time slot is the transmission time for an MPTCP packet from the base station. The channel state switches from a good state (all transmissions are successful) to a bad state (all transmissions are unsuccessful) and vice-versa.

Let c denote the maximum rate in packets/sec at which the base station can transmit, i.e., the bit rate ratio to the packet size in bits. With v as the velocity of the CAV, and f_c is the used carrier frequency, the Doppler frequency is $f_d = f_c v / v_c$, where v_c is the speed of light.

With F as the fading margin of the channel, if the received signal $< \mathbb{E}[\text{SINR}] / F$, the channel is in the bad (B) state; else, it is in the good (G) state; here, $\mathbb{E}[\text{SINR}]$ is the expected value of the received signal to interference noise ratio SINR. The transmission error probability due to a wireless channel error is $f_i = 1 - e^{-1/F}$, cf. [8]. Using [8]

$$\eta = \sqrt{2/(F(1 - \rho^2))}, \quad \rho = J_0(2\pi f_d/c),$$

where (i) ρ is the Gaussian correlation coefficient of a fading channel with frequency (Doppler) f_d , sampled t seconds away, and (ii) $J_0(\cdot)$ is the Bessel function of the first kind. The stationary channel state transition probabilities as shown in Fig. 2 can be computed as (see $\mathbb{Q}(\cdot, \cdot)$ the Marcum-Q-function [8], [30])

$$\begin{aligned} Pr(\text{Channel stays in B}) &= \bar{p}_b \\ &= \frac{\mathbb{Q}(\varrho\eta, \eta) + e^{\frac{1}{F}} - 1 - \mathbb{Q}(\eta, \varrho\eta)}{e^{\frac{1}{F}} - 1}, \quad \text{and} \end{aligned} \quad (9)$$

$$\begin{aligned} Pr(\text{Channel Stays in G}) &= \bar{p}_g \\ &= \frac{1 - f_i(2 - \bar{p}_b)}{1 - f_i}. \end{aligned} \quad (10)$$

Let ℓ_n denote the probability that at least one out of n link layer transmission frames is unsuccessful, given that the initial state is G. Here, R is the maximum number of retransmissions allowed, and L is the number of link layer frames in an MPTCP packet (TCP Acks fits into one link layer frame). Denoting $m_n^{(r)}$ as the probability that at least one out of n transmission frames is unsuccessful, given that the first link layer frame has already had r (with $r < R$) unsuccessful transmission attempts and the initial state is B. With the boundary conditions

$$\ell_1 = 0, \quad m_n^{(R)} = 1, \quad m_1^{(r)} = \bar{p}_b^{R-r}$$

, the recurrence relationships for these probabilities are

$$\begin{aligned} \ell_n &= \bar{p}_g \ell_{n-1} + (1 - \bar{p}_g) m_{n-1}^{(0)}; \\ m_n^{(r)} &= (1 - \bar{p}_b) \ell_n + \bar{p}_b m_n^{(r+1)} \end{aligned}$$

where the values of ℓ_n , $1 \leq n \leq L$ and $m_n^{(r)}$, $0 \leq r \leq R$ are obtained. Therefore, the probability of an MPTCP packet is loss over link i is given by

$$e_i = \frac{\ell_L(1 - \bar{p}_b) + m_L^{(0)}(1 - \bar{p}_g)}{(2 - \bar{p}_b - \bar{p}_g)}. \quad (11)$$

Using u_n as the mean number of link-layer frames required to be transmitted for an MPTCP data packet consisting of n link frames, given that the initial channel state is G. If $s_n^{(r)}$ is the mean number of link-layer frames required to be transmitted for an MPTCP data packet consisting of n link frames, the first link-layer frame has already had r (with $r < R$) unsuccessful transmission attempts and the initial state is B; by using boundary conditions $u_1 = 1$ and $s_1^{(R)} = 1$, the mathematical interpretation of the recurrence relations [8]

$$\begin{aligned} s_n^{(r)} &= 1 + \bar{p}_b s_n^{(r+1)} + (1 - \bar{p}_b) u_n \\ s_n^{(R)} &= 1 + \bar{p}_b s_{n-1}^{(0)} + (1 - \bar{p}_b) u_{n-1} \\ u_n &= 1 + \bar{p}_g u_{n-1} + (1 - \bar{p}_g) s_{n-1}^{(0)}. \end{aligned}$$

Finally, the expected delay for the transmission of an MPTCP data packet consisting of L link frames is

$$E[D] = \frac{u_L}{\left(\frac{c}{(1-\bar{p}_b)}(1 - \bar{p}_b + 1 - \bar{p}_g)\right)} + \frac{s_L^{(0)}}{\left(\frac{c}{(1-\bar{p}_g)}(1 - \bar{p}_b + 1 - \bar{p}_g)\right)}.$$

Observe from the above analysis that given the distribution of transmission time, the throughput of the memoryless channel can be computed along the lines of that [29]. Given the distribution of the delay, the average delay can be found by evaluating the first derivative as shown earlier (see [29] for details).

Remark III.1. In reality, the feedback over time-varying wireless channels is lossy and delayed, burst errors occur, and the round trip transmission time fluctuations can cause a high variability in the delay. To capture and quantify these effects, we explore and exploit the *three-sigma rule*, which has been known to be justifiable when the distribution of the delay can be approximated as sub-Gaussian. Given constants $a_1 > 0$, $a_2 > 0$, $Pr(D > d) < a_1 e^{-a_2 d^2} \quad \forall d > 0$, the distribution of D as a random variable is sub Gaussian. Such sub-Gaussian distribution has strong tail decay (at least as fast as the tails of a Gaussian). Therefore, the guaranteeable delay $D_{3\sigma}$ of URLLC-MPTCP is upper bounded by the guaranteeable delay of a Gaussian process with the same mean and deviations of D . It has been demonstrated [29] that *the tails of the Gaussian dominate the tails of the delay distribution*.

Considering that the delay distribution $\phi_D(z)$ is sub-Gaussian, the guaranteeable delay of the URLLC-MPTCP, given the expected delay $\mathbb{E}[D]$, can be computed as

$$\mathbb{E}[D_{3\sigma}] = \mathbb{E}[D] + 3\sigma \quad (12)$$

where σ^2 is the variance of the delay given by (see [29] for details)

$$\sigma^2 = \phi_D''(z)|_{z=1} + \mathbb{E}[D] - \mathbb{E}[D^2].$$

IV. RESULTS AND DISCUSSIONS

We performed comprehensive experiments to evaluate URLLC-MPTCP under several different network scenarios. Using network settings in our testbed, we have CAVs use-case scenario and settings, and then explain our observations based on findings and results. We evaluated URLLC-MPTCP by benchmarking it with established MPTCP protocols: BALIA [26], OLIA [25] and LIA [6], the celebrated *multipath congestion control* protocols. With relevant insights from [4], we have enhanced Paasch *et al.* Linux implementation⁶. The network setup consists of five laptops, one of them is a server, and four others are the user devices using Linux 16.04. All user machines are interconnected using a gigabit switch to the server using two gigabit Ethernet interfaces and links for our experiments.

The MPTCP connection from users to the server includes two different subflows in our experiments. We have used NetEm for bandwidth control, channel loss and latency for wireless links emulations.⁷ The experimental traces are evaluated by using *tcpdump*⁸ and quantified with *wireshark*.⁹ For generating the data traffic, we are using different size files from the server ranging from 2 Mega-Byte to 20 Mega-Byte; the data points in the plots are computed by extracting *tcpdumps*.

A. Delay Analysis and Accuracy

The predicted $\mathbb{E}[D]$ resequencing period obtained in Eqn. (8) offers a valuable estimate that can be used to evaluate the coding efficiency across several parallel wireless links. The assumption created in our study for coding and arrivals have been explained to be accurate with the tests.

Fig. 3. For both links, we use variable random link loss ($e_i = 3\%$) and equal file size (10 Mega-Byte) to observe the impact of resequencing delay with increased stream code rate perceived by the MPTCP client (different runs with URLLC-MPTCP with the same configuration).

For a URLLC-MPTCP connection using two links, a comparison of the reordering delay obtained from the model with experimental results for the coding scheme is given in Figures 3-4. Figure 3 reveals that the previously established estimate is a reasonably accurate measure of the true delay incur due to resequencing over a variety of code levels and packet losses (see [4] for other details). Comparing the latency

for communication with several concurrent links, the penalty for utilising multiple links is simply not very important. Nevertheless, we can observe that the future advantages of communication diversification through URLLC-MPTCP connections may be significant and invaluable. In Figure 3 the delay ratio for each connection is $d_1/d_2 = 50/20$, while in Figure 4 that ratio is $d_1/d_2 = 30/50$ Figure 4 also contains details about the variability of the delay due to the resequencing of packets and its variance.

For examples, the overall communication capacity using two links is approximately double that of communication using a single link only. Furthermore, each extra link increases the connection's reliability. It will help ensure low latency when CAVs networking is extremely intermittent or unstable network connections.

Fig. 4. For both links, we use varying random link loss and identical file size (10 Mega-Byte) to examine the impact of sequencing delay with increasing stream coding rates as obtained by the MPTCP users (separate passes with the same network configuration of URLLC-MPTCP protocol).

As discussed in [4], Figure 4 suggests that the delay is always primarily caused by the relation with the greater likelihood of packet loss magnitude. In fact, the delay isn't very prone to change in bandwidths (compare Figure 3 to Figure 4). Similar insights regarding reordering delay have already been reported in [4].

Fig. 5. For both links, we use varying random link loss and identical file size (10 Mega-Byte) to examine the impact on mean and guaranteeable delay with increasing channel loss rates as perceived by the MPTCP users (separate passes with the same network configuration of URLLC-MPTCP protocol).

With relevant insights from [29], we analyze our experimental results and observe in Figure 5 that both the mean

⁶"Multipath TCP in the Linux Kernel", <https://www.multipath-tcp.org> (released: 22. June 2019) of MPTCP v0.95

⁷<https://wiki.linuxfoundation.org/networking/netem>

⁸<https://www.tcpdump.org/>

⁹<https://www.wireshark.org/>

and guaranteeable delay of the coded and uncoded (set $R_i = 1$) URLLC-MPTCP increases with an increase in the loss rate e_i . Although the Coded MPTCP has a higher average delay than the uncoded MPTCP, the guaranteeable delay for Coded MPTCP is lower than the one for uncoded MPTCP. Furthermore, we observe that by increasing the timeout period, the separation between the guaranteeable delays can be reduced further.

B. Real WiFi Testbed Experiment

In actual WiFi networks, we performed an experiment with setting identical to previous studies but using two network interfaces (WiFi), where the measured two connection bandwidths were 54 Mega-bits per second and 10 Mega-bits per second. We have five MPTCP flows, and we calculate the expected output of each link. We perform the same experiment in the same way with all other multiple MPTCP protocols (different executions with *Ours*, *BALIA*, *LIA*, *OLIA*). Figure 6 shows the results and contrasts various MPTCP links perceived performance.

Fig. 6. Actual WiFi tests with one link bandwidth set at 54 Mega bits per second and another at 10 Mega bits per sec and 15 milli-seconds and 10 milli-seconds latency respectively. There we have five MPTCP connections, and we have calculated the expected individual connections throughput. In the same configuration, we conduct the same experiment using established MPTCP protocols—different tests with *BALIA*, *URLLC-MPTCP*, *LIA* and *OLIA*.

Under this situation (see caption of Figure 6), both the likelihood of packet failure and the latency are small (as opposed to previous test cases), the differences between our URLLC-MPTCP protocol and established MPTCP protocols are relatively limited (but noticeable).

C. Extensive Evaluation of URLLC-MPTCP and Findings

First, we set a capacity of 15 Mega bits per sec and delay 160 milli-seconds with a random loss of 2%, same for all links. Then we observe the throughputs and fairness using various

Fig. 7. Comparison of individual throughput with increasing file size in Megabytes.

Fig. 8. Throughput fairness (Jain's Index) with increasing file sizes (4 Mega-Byte $\leq$ size $\leq$ 8 Mega-Byte) from the MPTCP client.

sizes ranging from 4Mega – Byte $\leq$ size $\leq$ 8Mega – Byte using sever other MPTCP protocols for benchmarking our URLLC-MPTCP under the exact same network environment. Figures 7-8 illustrate the comparison results on throughputs and fairness. We observed that our URLLC-MPTCP protocol not only outperforms all existing MPTCP protocols in terms of throughputs but also achieve comparable fairness. We can see in Figure 7 when the size of the file is 4 Mega-Byte, proposed URLLC-MPTCP significantly outperforms *BALIA*, *LIA*, *OLIA*. Furthermore, observe that the decrease in throughputs with increasing file size is very small (due to stream coding) for 2% of packet transmission failures due to wireless errors. We have shown in Figure 8, that Jain's fairness indices of all MPTCPs, including URLLC-MPTCP, are quite close to *unity* (1) but the throughput with URLLC-MPTCP is significantly improved (without any loss on fairness). The explanation for this is the close coupling in implementing the proposed MPTCP to minimize irregular channel errors jointly (using stream coding) and reorder delay (using a load balancer, recall Eqn. (1))

For the next experiment, we use delay (160 milli-seconds) with (4%) likelihood of packet loss, same for all links and vary bandwidths of only one of the links (fixed another link bandwidth to 20 Mega bits per sec). We discovered the impact on expected throughputs using the same file of 10 Mega-Byte with several runs of other MPTCP protocols. This compares the proposed URLLC-MPTCP with others under the same network environment. Figure 9 illustrates the impact on throughputs.

The newly designed URLLC-MPTCP outperforms all other existing MPTCP protocols in terms of responsiveness and efficient utilization of asymmetric wireless links. The throughputs shown in Figure 9 perceived by seminal MPTCP protocols are relatively low and remains the same even with increasing bandwidths. Our new protocol shows improvements in resource pooling for the same time-varying network settings—the reason being that under erratic channel errors, traditional MPTCP protocols suffer a lot. However, due to inbuilt forward error correction with streaming code, URLLC-MPTCP combats pretty well even with significantly high packet loss due to wireless errors. The *balanced link adaptation* feature of URLLC-MPTCP protocol discovers asymmetric links shortly and performs balanced scheduling of packets across the links intelligently.

In another network scenario, we conduct an experiment and vary the delay of one of the links from 30 milli-seconds

Fig. 9. We consider varying bandwidths (one link bandwidth ranges from 20 mega bits per second to 100 mega bits per second while the other link bandwidth is fixed at 20 mega bits per second), the same latency (150 milli-seconds), the same random loss ($ei = 4\%$) and the same file size (10 mega-byte) on both links to analyze the effect of bandwidth variance on the expected (individual MPTCP connection) data throughput as experienced by the MPTCP users (using different MPTCPs *BALIA*, *URLLC-MPTCP*, *LIA*, *OLIA* under the exact same network environment).

to 60 milli-seconds (with a setting for other link delays to 20 milli-seconds) and set the same capacities and random (4%) likelihood of packet loss for both links so as to observe the impact on expected throughputs while downloading the same size (15 Mega-Byte) file using several MPTCP protocols (namely, *BALIA*, *URLLC-MPTCP*, *LIA*, *OLIA*) using the exact same network environment. Figure 10 shows how throughput varies.

Fig. 10. We consider varying delays (one link latency ranges from 20 milli-seconds to 100 milli-seconds while the other link latency is set at 20 milli-seconds), the same randomized loss ($ei = 3\%$) and the same file size (10Mega-Byte) for both links to examine the effect of variability in mean (individual connection) throughputs as interpreted by the MPTCP client (separate executes with *BALIA*, *URLLC-MPTCP*, *LIA*, *OLIA* but exactly same network environments).

The performance throughput obtained across all protocols (including Ours) decreases as predicted with increasing delay. But, for URLLC-MPTCP and others, the rate of decrease in the throughput is different. In other MPTCP protocols, the slope is very high, and for the current URLLC-MPTCP, it is fairly low. The explanation for this is the time spent discovering the delay spend by the other MPTCP protocols for the retransmission of missing packets and the resequencing of received packets at the user end (all transmitted packets must be received in order by MPTCP sinks). Our two new ideas which make URLLC-MPTCP adaptive and highly effective in severe the lossy environment is the decoupling of dependence (ordered delivery) by using streaming coded packets: When any of ω_i

(out of $\omega_i + \delta_i$ packets transmitted from sender) packets are received, why the need to wait for the missing packets in flight.

Next, we set the equal delay of 160 milli-seconds and capacity of 18 Mega bits per sec for both links but the time-varying likelihood of loss for one of the links (fixing other channel loss to 3%). We plot the impact on the expected throughputs while downloading the 10 Mega-Byte file while using all MPTCP protocols (several runs with *BALIA*, *URLLC-MPTCP*, *LIA*, *OLIA*) in the same network setting.

Fig. 11. We set time-varying loss (one link loss ranges from 3% to 5% whereas the other link loss is set constant at 3%), the same latency (150ms) and the same capacity (20 mega bits per second) when accessing the same file size (10mega-byte) utilizing both links to analyze the effect of loss on the expected (individual connection) performance as interpreted by the MPTCP client (several passes with *BALIA*, *LIA*, *OLIA* and *URLLC-MPTCP*) under the same environment).

Notice the effect on throughputs in the figure 11 – the URLLC-MPTCP is *resilient and performs resource pooling even under unfavourable channel conditions* as shown in Figure 11.

With increasing channel error likelihood in Figure 11 the expected throughputs from the established MPTCP protocols (*BALIA*, *LIA*, *OLIA*) is decreasing in nature. URLLC-MPTCP provides reliable performance, even in this state, with increasing channel errors and the rate of decrease is slower. This is because all established protocols obey a predetermined fixed congestion control policy. In particular, the existing solutions are quite conservative and substantially reduce their congestion windows only after identifying packet losses. This phenomenon incurs long delays (round trip time where many hundreds of packets in flight often suffer).

Nevertheless, our URLLC-MPTCP has the potential to absorb certain degrees of variance or channel fluctuations (using stream coding) and takes a few rapid attempts to learn the best way to adjust the congestion windows (using the load balancer).

One more possibility is known where a channel is dropped unexpectedly and became unavailable (for example, WiFi communication (small coverage area) often fails after 50 seconds when 5G is still on sale in freeways). The findings are illustrated in Figure 12, where the understanding of the output of irregular subflows from various MPTCP flows is contrasted with URLLC-MPTCP.

In response to the variation in the number of subflows, our URLLC-MPTCP functions relatively better, overflowing with the current MPTCP protocols, *BALIA*, *LIA*, *OLIA*.

Fig. 12. High-speed motorways comprising of roadside units of both 5G and WiFi links, but after 50 seconds, the WiFi link is lost. With these same settings, we first maintained five MPTCP connections and lost one of the subflows of each connection at 50 seconds during the test interval of 300 seconds (performed by shutting the interface off). In the very same setting, we run the same experiment using different MPTCP protocols (separate passes with BALIA, URLLC-MPTCP, LIA, OLIA)

Nonetheless, unlike all other MPTCP protocols, our URLLC-MPTCP protocol discovers a drastic decrease in the performance of any subflow over time, owing to the shift in the practical resource utilization scenario. The underlying mechanism induces the lack of flows very easily to initiate our congestion management mechanism (notice that Eqn. (1)'s equilibrium factor with $d_i = 0$ is zero, so MPTCP sender can easily locate the missing flow) and removes the subflow window. This process enables our protocol to be more sensitive than existing MPTCP protocols to changes in network links.

V. URLLC-MPTCP: CONVERGENCE AND STABILITY

The scientific theory of fluid movement has been praised for designing and analysing the protocols TCP and MPTCP [26]. This method combines short-term ergodic variations in value quantities such as delays and congestion times and also supports conventional differential equations to describe the complexities of packet transmission.

With applicable observations from the Peng *et al.* [26] multipath formulations, $k_i(\mathbf{x})$ is a vector of benefits (say) evaluating the functional properties of URLLC-MPTCP and $\phi_i(\mathbf{x})$ evaluating its equilibrium conditions. The required fluid model for the proposed URLLC-MPTCP for a given packet sending rate $x_i = \left(\frac{\omega_i/R_i}{\tau_i}\right)$ can be expressed as (see Section II.B and Eqn. (1)) [4]

$$\dot{x}_i = k_i(\mathbf{x})(\phi_i(\mathbf{x}) - \mathbf{e})_{x_i}^+ \quad (13)$$

$$\dot{e}_i = m_i(x_i - R_i/d_i)_{e_i}^+, \quad m_i > 1 \quad (14)$$

for investigating the uniqueness, convergence and existence of the developed MPTCP protocol. The packet sending rates $\mathbf{x} = (x_1(t), \dots, x_n(t))$, link loss $\mathbf{e} = (e_1(t), \dots, e_n(t))$, subscripts, $1, \dots, n$, represent the number of links respectively, and

$$z_y^+ = \begin{cases} z, & y > 0 \\ \max(z, 0), & y = 0. \end{cases}$$

Then the two links network dynamics under the URLLC-MPTCP protocol and Eqn. (1) can be modeled by Eqn. (13) where $i \in \{1, \dots, n\}$

$$k_i(\mathbf{x}) = \bar{k}_i(x_i) = x_i^2/2, \quad (15)$$

$$\phi_i(\mathbf{x}) = \frac{\max_{j \in \{1, \dots, n\}} \{x_j/\tau_j\}}{x_i \tau_i \sum x_i^2} 2\beta_i. \quad (16)$$

Next, it is essential to study the stability of the data flow dynamic system under our URLLC-MPTCP protocol $\mathbf{k}$ and ϕ and their dynamic properties (recall Eqn. (1)). We use the following four conditions for the tractability [4], [8], [26].

- COND.1: Continuous and symmetric Jacobians $\partial\phi(\mathbf{x})/\partial\mathbf{x}$.

As known for MPTCP protocols, our URLLC-MPTCP protocol fails to satisfy COND.1; therefore, the utility function does not exist.

- COND.2: $\mathbf{x} := \mathbf{x}(\mathbf{e})$ exists with

$$\frac{\partial x_i}{\partial e_i} \leq 0, \quad \lim_{e_i \rightarrow \infty} x_i(\mathbf{e}) = 0.$$

- COND.3: Continuously differential (negative definite) $\phi(\mathbf{x})$.
- COND.4: $\phi_i(\mathbf{x}) = \infty$ iff packet rate $x_i = 0$ from the URLLC-MPTCP sender to the link i .

When COND.2 holds, Eqn. (13) with the URLLC-MPTCP protocol Eqn. (1) has a minimum of one equilibrium. Also, when both COND.3 and COND.4 hold together, Eqn. (13) has a maximum of one equilibrium. In essence, for the three conditions: COND.2, COND.3 and COND.4 to hold, Eqn. (13) has only one unique equilibrium $(\mathbf{x}^*, \mathbf{e}^*)$. Further details on the convergence analysis is provided in the appendix.

VI. CONCLUSION

We have extended the multipath TCP protocol for URLLC and evaluated the new URLLC-MPTCP for low latency applications such as connected autonomous vehicles. The newly developed MPTCP algorithm has adopted the stream coding policy, which allows efficient use of wireless connection through several heterogeneous parallel networks, thus meeting the guaranteeable latency constraints of the vehicles. Our experimental results show that this novel URLLC-MPTCP learns evolving network constraints and adapts to the time-varying system dynamics. We developed an approximation for guaranteeable delay and resequencing delay of URLLC-MPTCP protocol at the receiver end and demonstrate that URLLC-MPTCP effectively supports latency-sensitive applications. In conclusion, the development towards URLLC-MPTCP has been able to harness bandwidth from heterogeneous networks and facilitates seamless experience over CAVs network, which requires further enhancements.

APPENDIX

A. Complexity of URLLC-MPTCP

Our analysis in this section involves identifying the order in which paths become bottlenecked. Let

$$\mathcal{I}(S) = \max_j \in S \left\{ \frac{\sqrt{\omega_j}}{\tau_j} \right\} / \sum_j \frac{\omega_j}{\tau_j} \quad (17)$$

and the fractions, when kept in order, are

$$\left\{ \frac{\sqrt{\omega_1}}{\tau_1} \right\} \leq \left\{ \frac{\sqrt{\omega_2}}{\tau_2} \right\} \leq \left\{ \frac{\sqrt{\omega_3}}{\tau_3} \right\} \leq \dots \left\{ \frac{\sqrt{\omega_n}}{\tau_n} \right\}. \quad (18)$$

It is worth observing that with this ordering in (18), the equilibrium window increase of (1) (in Step 6 Algorithm 1) can be computed as [17]

$$\min_{S \subset R: j \in S} \frac{\frac{\omega_{\max}(S)}{\tau_{\max}^2(S)}}{\left(\sum_{s \in S} \frac{\omega_s}{\tau_s}\right)^2} = \min_{j \leq u \leq n} \frac{\frac{\omega_u}{\tau_u^2}}{\left(\sum_{t \leq u} \frac{\omega_t}{\tau_t}\right)^2} \quad (19)$$

i.e., it can be computed with a linear search.

B. Convergence of URLLC-MPTCP

The following derivation and mathematical analysis characterize the stability and convergence of network data flow dynamics under our URLLC-MPTCP (see Eqn. (1)). The conditional existence and uniqueness of $(\mathbf{x}^*, \mathbf{e}^*)$ is provided in the following, where conditions COND.2, COND.3 and COND.4 are given. With relevant insights from [4], [26], we construct the following lemma.

Lemma VI.1. *From an arbitrary initial point $(\mathbf{x}(0), \mathbf{e}(0))$, the continuous moving path vector $(\mathbf{x}(t), \mathbf{e}(t))$ traversed by the developed URLLC-MPTCP stabilizes to a globally converging equilibrium point $(\mathbf{x}^*, \mathbf{e}^*)$ iff COND.2, COND.3 and COND.4 hold true.*

Using $\partial \mathbf{x} := \mathbf{x} - \mathbf{x}^*$ and $\partial \mathbf{e} := \mathbf{e} - \mathbf{e}^*$ we construct the Lyapunov function

$$\mathcal{V}(\mathbf{x}, \mathbf{e}) = \sum_{i \in \{1, \dots, n\}} \left[\int_{x_i^*}^{x_i} \left(\frac{z - x_i^*}{\bar{k}_i(z)} \right) dz + \frac{1}{2} \frac{\partial e_i^2}{m_i} \right]. \quad (20)$$

The proof of Lemma VI.1 follows along the lines of that of [8, Thm. 5.4.2 p.141], however, we include the essential steps as follows (see [26] for details).

Proof. Differentiating $\mathcal{V}(\mathbf{x}, \mathbf{e})$ with respect to t is

$$\dot{\mathcal{V}}(\mathbf{x}, \mathbf{e}) = \sum_{i \in \{1, \dots, n\}} \left(\frac{\partial e_i \dot{e}_i}{m_i} + \frac{1}{\bar{k}_i(x_i)} \partial x_i \dot{x}_i \right). \quad (21)$$

Now we use Eqn. (14) for $\partial \mathbf{x} \neq 0$

$$\begin{aligned} \frac{\partial x_i \dot{x}_i}{\bar{k}_i(x_i)} &= \partial x_i (\phi_i(\mathbf{x}) - e_i)_{x_i}^+ \\ &\leq \partial x_i (\phi_i(\mathbf{x}) - e_i) \\ &= \partial x_i (\phi_i(\mathbf{x}) - \phi_i(\mathbf{x}^*) - \partial e_i). \end{aligned} \quad (22)$$

and

$$\begin{aligned} \sum_{i \in \{1, \dots, n\}} \left(\frac{1}{\bar{k}_i(x_i)} \partial x_i \dot{x}_i \right) &\leq \partial \mathbf{x}^T (\phi(\mathbf{x}) - \phi(\mathbf{x}^*)) - \partial \mathbf{x}^T \partial \mathbf{e} \\ &< -\partial \mathbf{x}^T \partial \mathbf{e}. \end{aligned} \quad (23)$$

Now, first part of Eqn. (21),

$$\begin{aligned} \sum_{i \in \{1, \dots, n\}} \frac{1}{m_i} \partial e_i \dot{e}_i &= \sum_{i \in \{1, \dots, n\}} \partial e_i \left(x_i - \frac{R_i}{d_i} \right)_{e_i}^+ \\ &\leq \sum_{i \in \{1, \dots, n\}} \partial e_i \left(x_i - \frac{R_i}{d_i} \right) \leq \sum_{i \in \{1, \dots, n\}} \partial e_i \partial x_i = \partial \mathbf{e}^T \partial \mathbf{x}. \end{aligned} \quad (24)$$

Then,

$$\begin{aligned} \text{for } \partial \mathbf{x} \neq 0, \quad \dot{\mathcal{V}}(\mathbf{x}, \mathbf{e}) &< \partial \mathbf{e}^T \partial \mathbf{x} - \partial \mathbf{x}^T \partial \mathbf{e} = 0; \\ \text{for } \partial \mathbf{x} = 0, \quad \dot{\mathcal{V}}(\mathbf{x}, \mathbf{e}) &= 0. \end{aligned} \quad (25)$$

Using Lasalle's invariance, $(\mathbf{x}^*, \mathbf{e}^*)$ is a stable global equilibrium. The reason being that the only path in vector $\mathbb{P} := (\mathbf{x}(t), \mathbf{e}(t))$ is the path $(\mathbf{x}, \mathbf{e}) \equiv (\mathbf{x}^*, \mathbf{e}^*)$; because this is only the path vector $\mathbb{P}$ on which $\dot{\mathcal{V}}(\mathbf{x}(t), \mathbf{e}(t)) = 0$ for all $t \geq 0$. $\square$

REFERENCES

- [1] S. R. Pokhrel, N. Kumar, and A. Walid, "Towards ultra reliable low latency multipath tcp for connected autonomous vehicles," *IEEE Transactions on Vehicular Technology*, accepted, 2021.
- [2] M. Bennis, M. Debbah, and H. V. Poor, "Ultrareliable and low-latency wireless communication: Tail, risk, and scale," *Proceedings of the IEEE*, vol. 106, no. 10, pp. 1834–1853, 2018.
- [3] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE network*, vol. 34, no. 3, pp. 134–142, 2019.
- [4] S. R. Pokhrel and J. Choi, "Low-delay scheduling for internet of vehicles: Load-balanced multipath communication with FEC," *IEEE Transactions on Communications*, vol. 67, no. 12, pp. 8489–8501, 2019.
- [5] S. R. Pokhrel, J. Jin, and H. Le Vu, "Mobility-aware multipath communication for unmanned aerial surveillance systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 6088–6098, 2019.
- [6] C. Raiciu, M. Handley, and D. Wischik, "Coupled congestion control for Multipath Transport Protocols, IETF RFC 6356," 2011.
- [7] S. Rene, E. Expósito, M. Gineste, J. Alins, and O. Esparza, "Multipath TCP Architecture for Infotainment Multimedia Applications in Vehicular Networks," in *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*. IEEE, 2015, pp. 1–5.
- [8] S. R. Pokhrel and M. Mandjes, "Improving Multipath TCP Performance over WiFi and Cellular Networks: an Analytical Approach," *IEEE Transactions on Mobile Computing*, 2018.
- [9] H. Chen *et al.*, "Ultra-reliable low latency cellular networks: Use cases, challenges and approaches," *IEEE Communications Magazine*, vol. 56, no. 12, pp. 119–125, 2018.
- [10] S. R. Pokhrel and J. Choi, "Improving TCP performance over WiFi for internet of vehicles: A federated learning approach," *IEEE Transactions on Vehicular Technology*, 2020.
- [11] 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. 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 2012, pp. 29–29.
- [12] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental Evaluation of Multipath TCP Schedulers," in *Proc. 2014 ACM SIGCOMM workshop on Capacity sharing*. ACM, 2014, pp. 27–32.
- [13] Y.-C. Chen and D. Towsley, "On Bufferbloat and Delay Analysis of Multipath TCP in Wireless Networks," in *Proc. Networking, IFIP*. IEEE, 2014, pp. 1–9.
- [14] S. R. Pokhrel, H. L. Vu, and A. L. Cricenti, "Adaptive admission control for IoT applications in home WiFi networks," *IEEE Transactions on Mobile Computing*, 2019.
- [15] S. R. Pokhrel and C. Williamson, "A rent-seeking framework for multipath TCP," in *IFIP 2020, PERFORMANCE*, 2020, pp. 00–00.
- [16] Y. Cui, L. Wang, X. Wang, H. Wang, and Y. Wang, "FMTCP: A Fountain Code-based Multipath Transmission Control Protocol," *IEEE/ACM Transactions on Networking*, vol. 23, no. 2, pp. 465–478, 2015.
- [17] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, "Design, Implementation and Evaluation of Congestion Control for Multipath TCP," in *USENIX, NSDI*, 2011, pp. 280–288.
- [18] S. R. Pokhrel, J. Ding, J. Park, O.-S. Park, and J. Choi, "Towards enabling critical MMTC: A review of URLLC within MMTC," *IEEE Access*, vol. 8, pp. 131 796–131 813, 2020.
- [19] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the Air: Practical Wireless Network Coding," in *ACM SIGCOMM computer communication review*, vol. 36, no. 4. ACM, 2006, pp. 243–254.

- [20] J. Cloud and M. Médard, "Multi-path Low Delay Network Codes," in *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016, pp. 1–7.
- [21] D. J. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge university press, 2003.
- [22] F. Kelly and T. Voice, "Stability of End-to-End Algorithms for Joint Routing and Rate Control," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 2, pp. 5–12, 2005.
- [23] T. Voice, "Stability of Multi-path Dual Congestion Control Algorithms," in *Proc. 1st international conference on Performance evaluation methodologies and tools*. ACM, 2006, p. 56.
- [24] H. Han, S. Shakkottai, C. Hollot, R. Srikant, and D. Towsley, "Multi-path TCP: A Joint Congestion Control and Routing Scheme to Exploit Path Diversity in the Internet," *IEEE/ACM Transactions on Networking*, vol. 14, no. 6, pp. 1260–1271, 2006.
- [25] R. Khalili, N. Gast, M. Popovic, and J.-Y. Le Boudec, "MPTCP is not Pareto-optimal: Performance Issues and a Possible Solution," *IEEE/ACM Transactions on Networking*, vol. 21, no. 5, pp. 1651–1665, 2013.
- [26] Q. Peng, A. Walid, J. Hwang, and S. H. Low, "Multipath TCP: Analysis, Design, and Implementation," *IEEE/ACM Transactions on Networking*, pp. 596–609, 2016.
- [27] A. Walid, Q. Peng, J. Hwang, and S. Low, "Balanced linked adaptation congestion control algorithm for MPTCP," 2014.
- [28] Y.-C. Chen, Y.-s. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, and D. Towsley, "A measurement-based study of multipath TCP performance over wireless networks," in *Proc. Internet Measurement Conference*. ACM, 2013, pp. 455–468.
- [29] D. Malak, M. Médard, and E. M. Yeh, "Tiny codes for guaranteeable delay," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 4, pp. 809–825, 2019.
- [30] M. Zorzi, R. R. Rao, and L. B. Milstein, "ARQ error control for fading mobile radio channels," *IEEE Transactions on Vehicular Technology*, vol. 46, no. 2, pp. 445–455, 1997.