Novel Multipath TCP Scheduling Design for Future IoT Applications

Mohammed Asiri, masiri@deakin.edu.au, Deakin University, Geelong, VIC

Abstract

Today, mobile devices like smartphones are supported with various wireless radio interfaces including cellular (3G/4G/LTE) and Wi-Fi (IEEE 802.11) [46]. The legacy devices can only communicate with only one interface. The Transmission Control Protocol, or TCP, has a limitation inability to change connection settings without breaking the connection. Multi-path TCP (MPTCP) protocol has been proposed to solve TCP single-interface limitation and provides a huge improvement on application performance by using multiple paths transparently (auto path changing). The last mile is the final networking segment which carried all network traffic. The available bandwidth in last-mile link can be effectively harms the network throughput as it limits the amount of transmitted data. The quality of the last mile networks significantly determines the reliability and quality of the carrying network. MPTCP can provide a convenient solution for the last mile problem. A MPTCP scheduler needs to provide significant packet routing schedules based on the current status of paths (sub-flows) in terms of loss rate, bandwidth and jitter, in a way, maximizing the network goodput. MPTCP extends the TCP by enabling single byte stream split into multiple byte streams and transfer them over multiple disjoint network paths or subflows. A MPTCP connection combines a set of different subflows where each subflow performance depends on the condition of its path (including packet loss rate, queue delay, and throughput capacity). Unreliable packet scheduling may lead to critical networking issues such as the head-of-line (HoL) blocking where the packets scheduled on the low-latency path must wait for the packets on the high-latency path to ensure in-order delivery and the out-of-order (OFO) packets, the receiver must maintain a large queue to reorganize the received packets. In this project, we aim to study and experiment MPTCP scheduling on dynamic networks (like cellular network) and try to propose a MPTCP schema which can be effective to overcome limitations of dynamic networks performance.

1 Introduction

Today, mobile devices smartphones and tablets are supported with multi-wireless communication interfaces such as cellular (3G/4G/LTE) and Wi-Fi (IEEE 802.11). The transmission control protocol (TCP) is the standard communication schema in current computer network systems. TCP protocol was originally designed for a single node-to-node and once a connection is established, it's elements cannot be changed without breaking the connection. Such elements are: sender IP address and port, and receiver IP address and port. However, TCP does not benefit from the multi-interface capability and mobility of mobile devices and thus limits the performance multiple routes usage in case of non-stable network conditions. A potential solution is the adoption of Multi-path TCP (MPTCP) to improve application performance by using multiple paths transparently

(auto path changing). Pokhrel et al. [43] found that wireless channel errors and buffer overflows have negative impact on achieving fair throughput over TCP links.

MPTCP is an extension of the TCP which allows mobile devices to use various network interfaces simultaneously. Figure 1 shows atypical MPTCP architecture. The core idea of MPTCP is to split a single byte stream to multiple byte streams and transfer them over multiple disjoint network paths, so called subflows. The MPTCP is prominent to enhance communication network throughput and achieve robustness [15]. The quality of a subflow is determined by its connection path including signal converge, loss rate, queue delay, performance of involving links and connectivity of access points [25, 4]. To manage the distribution of data packet over heterogeneous subflows, a packet scheduler is employed to select the optimized scheduling plan based on various objectives such as reducing transmission delay, reducing communication cost and increasing network throughput, as shown in Figure 2. The scheduling process implies determining the amount of data packets to be distributed onto the subflows, which has a significant impact on the performance of MPTCP. The core idea of MPTCP scheduling optimization is to shift all data traffic onto the least-congested path. This called multipath congestion control procedure [53]. For a flow connection, MPTCP should provide throughput as minimum of SPTCP (single-path TCP) on the high-quality paths and no more capacity on any path or collection of paths in SPTCP. In context of IoT applications, MPTCP scheduling have challenges to improve performance such as scalability, reliability and latency [37].

Multi-path TCP (MPTCP) has the potential to greatly improve application performance by using multiple paths transparently (auto path changing). MPTCP was proposed as an extension of the TCP which enables multihomed devices to make simultaneous use of various network interfaces. The key idea of MPTCP (Multipath TCP) is to split a single byte stream to multiple byte streams and transfer them over multiple disjoint network paths. It adds path diversity to a traditional TCP in order to

Figure 1: Multi-path TCP architecture [2]

expedite throughput and achieve robustness [15]. Network communication interfaces are heterogeneous in nature based on the variation on communication quality such as signal converge, loss rate, performance of involving links and connectivity of access points [25, 4]. A MPTCP connection consists of a set of different subflows, whose individual performance relies on the condition of its path. The path condition is determined by the state of its bottleneck link, such as packet loss rate, queue delay, and throughput capacity. To deal with multipath diversity, a scheduler has to select the best available subflow to send each packet.

Packet scheduling is a unique and fundamental mechanism for the design and implementation of MPTCP. A scheduler is responsible for determining the amount of data packets to be distributed onto the subflows, which has a significant impact on the performance of MPTCP. In the context of heterogeneous networks where packets are multiplexed across multiple paths with wide delay difference (e.g., WiFi and LTE), improper packet scheduling may cause: 1) the head-of-line (HoL) blocking [49]: the packets scheduled on the low-latency path have to wait for the packets on the high-latency path to ensure in-order delivery. 2) To accommodate the out-of-order (OFO) packets, the receiver must maintain a large queue to reorganized the received packets [31]. Requirements of MPTCP scheduler can be summarized as: 1) It should consider network heterogeneity (e.g., delay and capacity), 2) It must balance a variety of QoS goals such as maximizing average throughput, reducing the overall data transfer time, minimizing self-inflicted latency or out-of-order buffer size, and maintaining jitter smoothness and 3) It should adapt to the dynamic of network environments.

Section 2 reviews the literature. Section 3 presents the research design and methodology. Section 4 describes the approach and the technical details of artefact development. Section 5 evaluates the artefacts, on the basis of research questions (RQs) in Section 5.1 and discusses the RQs in Section 6. Section ?? discusses threats to validity

Figure 2: Multipath scenario should consider the path heterogeneity [54]

and Section 7 concludes the report.

2 Literature Review

Heterogeneous networks, such as in IoV systems, bring challenges for achieving high performance MPTCP scheduling. In 4G/LTE networks, the connection path uses large buffers with long fluctuating in transmission, with high delays and packet loss rates. On the other hand, in case of WiFi path, its short delays and higher packet loss rates. The network performance is affected by various parameters such as buffer sizes at data receivers, queuing, number of flows sharing the connection and others. In IoV networks, which rely on WiFi networks, the wireless medium is shared by connections causing issues like repeated handoffs and network collisions. In general, IoV applications share the requirements of throughput-intensive (low end-to-end delay) and high demanded for reliable data delivery. Thus, wide range of MPTCP protocols and techniques focus on designing TCP congestion control schemes to maximize network throughput.

MultiPath TCP (MPTCP) is a modified version of TCP protocol which provides guidelines to distribute data packets over several subflows to maximize resource utilization and throughput. Generally, a successful MPTCP scheduler should be able to: provides reliable scheduling while meeting network heterogeneity (such as delay, capacity and loss rate), 2) balance a variety of QoS constraints (such as maximizing throughput, reducing RTT and minimizing latency) and 3) adaptive to network dynamicity (real time network conditions). For example, the high fluctuation of cellular and Wi-Fi networks performance [51].

2.1 MPTCP: Packet scheduling

MPTCP has certain challenges to achieve high throughput packet delivery. Firstly, the Head-of-line blocking, or shortly HOL-blocking, which is a networking state performance-limiting phenomenon that takes place when a sequence of packets is blocked by the first packet [49]. In HOL-blocking state, packets scheduled on the faster subflows arrive at destination buffer and then wait for the arrival of the packets scheduled on the slower path. This cause two main issues, high-congestion at the receiver buffer and out-of-order(OFO) packet delivery. The out-of-order(OFO) issue also can be occurred with high packet loss and drop rate subflows. One solution for OFO is to maximize the waiting queue at the receive side [31]. However, employing large buffers (queues) can cause bufferbloat issue where packets remain enqueued for a long time, particularly with under congestion paths. These issues are drastically decrease the network performance (in term of throughput), expressly for delay-sensitive applications in IoT and IoV application contexts [1]. To conclude, to achieve a satisfied level of MPTCP performance, both congestion control policy and packet scheduler should be convenient to align the requirements of the networking system environment.

According to Raiciu et al. [48], the core concept of MPTCP is to schedule packets to reduce the traffic on high congested paths through controlling the congestion window. This can be obtained by tuning the congestion window size to achieve high throughput

packet distribution over multiple paths (packet scheduling). The last mile is the final networking segment which carried all network traffic. The available bandwidth in last-mile link can be effectively harms the network throughput as it limits the amount of transmitted data. Another issue is the single point of failure which might cause whole network failure. Thus, MPTCP can be convenient to resolve last-mile link issues and achieve the desirable network performance QoS. Moreover, Pokhrel et al. [42] observed throughput unfairness of MPTCP with regular in last-mile WiFi networks.

Congestion control is an algorithm to adapt the congestion window (CWND) within subflows to control the data transfer rate at each subflow [30]. A congestion control should satisfy three rules. The first rule is "Improving Throughput", which implies that each multi-path flow performance should be at least as a single path flow performance. The second rule "Do not Harm", in which, a multi-path flow uses an equal capacity of the shared resources. The last rule is "Balance Congestion" which implies that a multi-path flow should transmit traffic like the most congested paths. There are two types of MPTCP congestion control algorithms, uncoupled and coupled. In the uncoupled congestion control, the simplest form, such as in EWTCP [19], each subflow is treated like an independent TCP connection. This congestion control algorithm does not satisfy fairness rule of MPTCP (rules 1 and 2) [56]. MPTCP cannot apply the standard TCP control scheme without unfair to normal TCP flows [6]. To resolve the unfairness issue, all congestion windows of all subflows are coupled. This allows an adaptive process to maintain the congestion window (CWND) of each subflow. Fully Coupled [18] and LIA [22] are examples of MPTCP coupled congestion controls. Barré et al. [6] provide a structured design framework on how to implement the MPTCP in the Linux kernel in order provide performance analysis to demonstrate that coupled congestion control is fairer than the standard one in TCP. Pokhrel et al. [40] shows that the coupled MPTCP LIA greatly suffers from the competition with TCP because of the reordering delay at the receiver. The next section provide a review of MPTCP scheduling techniques.

2.2 MPTCP: Packet scheduling

Congestion control is an algorithm to adapt the congestion window (CWND) within subflows to control the data transfer rate at each subflow [30]. A congestion control should satisfy three rules. The first rule is "Improving Throughput", which implies that each multi-path flow performance should be at least as a single path flow performance. The second rule "Do not Harm", in which, a multi-path flow uses an equal capacity of the shared resources. The last rule is "Balance Congestion" which implies that a multi-path flow should transmit traffic like the most congested paths. There are two types of MPTCP congestion control algorithms, uncoupled and coupled. In the uncoupled congestion control, the simplest form, such as in EWTCP [19], each subflow is treated like an independent TCP connection. This congestion control algorithm does not satisfy fairness rule of MPTCP (rules 1 and 2) [56].

MPTCP cannot apply the standard TCP control scheme without unfair to normal TCP flows [6]. To resolve the unfairness issue, all congestion windows of all subflows are coupled. This allows an adaptive process to maintain the congestion window (CWND) of each subflow. Fully Coupled [18] and LIA [22] are examples of MPTCP coupled congestion controls. Barré et al. [6] provided a structured design framework on how

to implement the MPTCP in the Linux kernel in order to analyze the contribution of coupled congestion control as a fairer packet transmission technique than the standard one in TCP. However, for MPTCP, the implementation of coupled strategy is challenging due the reordering delay at the data receiver side [40]. The next section provide a review of MPTCP scheduling techniques.

2.2.1 Baseline schedulers

Currently, the Linux kernel [48] (LIA) is the standard implementation of MPTCP. LIA applies the coupled congestion control to for adaptive congestion window calculation in consideration to various properties such as workload distribution fairness, data communication responsiveness, and congestion balance). In comparison to standard TCP (using only Wi-Fi), results show a throughput increases by 50% to 100% under various Wi-Fi coverage. Khalili et al. [23] reported that (LIA) is not Pareto-optimal and violates the fairness goal of MPTCP (achieving both fairness and responsiveness). They developed an improved version of LIA, called opportunistic linked-increases algorithm (OLIA), which tries to estimate the congestion window in an additive model while applies standard TCP behavior in the case of packet loss. They concluded on the performance dependency (for throughput) between subflows as it cannot increase the throughput of one subflow without decrease the throughput or increase the congestion cost. Peng et al. [32] proposed a fluid model called, Balanced Adaptive LIA (BALIA) presented comprehensive analysis for designing MPTCP and proposed balanced linked adaptation (Balialia) as a generalization of MPTCP congestion control algorithms. Result show a good balance in terms of TCP-friendliness and responsiveness.

Pokhrel et al. [40] proposed an analytical algorithm to improve the coupled congestion-based approach, in term of goodput, which is a network throughput measurement at the application-level (only concern about useful data transmission). The proposed approach works by by controlling the reordering delay at the receiver side, with consideration to packet loss rate and data transmission delay. Results show an improved goodput under constraint of low variation data rate.

2.2.2 Heuristics-based schedulers

The literature shows alternative strategies for MPTCP scheduling, apart from congestion window control based. MinRTT [24] is based on Round-Trip Time (RTT), which is the time difference between sending a data packet (with a particular sequence number) and receiving an acknowledgment packet (covers that sequence number) [47]. Subflows with lowest CWD are firstly selected before the ones with with higher RTTs. MinRTT technique has a limitation of HoL-blocking in heterogeneous networks as it mainly concerns about utilizing Congestion Window (CWND) and does not provide a pre-estimation of number of packets over available paths. Ferlin et al. proposed BLEST (BLocking ESTimation) [14] which distributes packets based on the minimum risk of HoL-blocking. It causes under-utilization of subflows, which may lead to download time inflation in case of transmitting larger data files. The proposed scheduler shows an improvement of 12% in application goodput while deducting the percentage of unnecessary retransmissions by 80% in comparison to default MPTCP scheduler.

Lim et al. [26] proposed ECF (Earliest Completion First) examine whether the default MPTCP path scheduler can provide applications the ideal aggregate bandwidth, i.e., the sum of available bandwidths of every paths. Experimental results show that heterogeneous paths cause underutilization of the fast path, resulting in undesirable behaviors for a real-time streaming application quality. The ECF consistently utilizes all available paths more efficiently than other approaches under path heterogeneity, particularly for streaming applications. Guo et al. [16] reported that high buffer size causes HoL-blocking with large data files. They proposed DEMS (DEcoupled Multipath Scheduler) which aims to reduce the data transfer time through an adaptive data redundant transmission (for small data chunks) over multi-paths. Adarsh et al. [2] argue that the main limitation of the default MPTCP scheduler is only the consideration of the RTT. However, this is not the prefect scheduler in case of heterogeneous subflows; matrices like performance and loss rate should be also considered. Pokhrel et al. [34] proposed a novel throughput-based MPTCP algorithm to schedule data packets in time-varying heterogeneous wireless paths. The algorithm targets Internet of Vehicle (IoV) systems, delay-sensitive networks, and applied a jointly technique of load balancing and forward error correction (FEC) for performing coupled congestion control MPTCP. The proposed technique outperforms traditional congestion control model through intelligent congestion control coupling with load balancing IoV networks.

Pokhrel and Garg [38] observed that optimization based MPTCP techniques will not be sufficient for high dynamic networks in terms of model complexity and real-time scheduling accuracy. The proposed learning-based scheduling can overcome these challenges through enabling the source to learn the best way to control itself from its own experience.

2.2.3 Learning-based schedulers

Heterogeneous paths, such as in cellular networks, are highly variable and unpredictable channels in terms of self-inflicted queuing delays and loss rate. This motivates the research towards adaptive scheduling techniques which consider the variability in network conditions while optimizing objectives like reducing delay and loss rate and increasing throughput. Learning-based schedulers are proposed to achieve these objectives through applying machine learning and deep learning techniques to understand network behaviour and generate efficient scheduling policies accordingly. Pokhrel and Singh [45] designed a novel analytical model to evaluate the performance of the coordinated TCP (C-TCP) flows for IoT applications over WiFi networks using a federated learning (FL) technique.

Chung et al. [10] proposed a novel path management scheme called MPTCP-ML which controls path usage based on machine learning mechanism. The objective is to periodically evaluate the quality of active paths at real-time based on patterns collected from heuristics. various quality metrics included such as throughput, RTT, signal quality and data rate. The random forests model was used to extract these patterns. Results show high prediction accuracy in mobile environment. Beig et al. [7] throughput-based learning algorithm to find the best signal quality rate in respect to interface type and transmitted file size. Three techniques examined: LTE, WiFi and MPTCP fro mobile devices. The performance evaluation shows MPTCP throughput improvement by 10% compared SPTCP over LTE and WiFi. However, the model does not consider other parameters

like loss rate and buffer size. Pokhrel and Mandjes [40] developed a comprehensive technique to evaluate the performance of durable (mostly permanent) MPTCP flows over heterogeneous network (WiFi and Cellular network). They considered various network features like retransmission limit and the buffer sizes. They proposed a learning algorithm to exploit the route heterogeneity as well as study the influence of heterogeneity variation on over all MPTCP performance. The coexistence of heterogeneous networks can be achieved by MPTCP through dynamic subflows management (to add and drop) according to according to certain QoS requirements in a best-effort manner. Such applications are in-city railway [27] and coordinating the convoy of drones [39]. Chiariotti et al. [9] worked on delay-sensitive MPTCP systems under user-defined constraint. They proposed a QoS-based MPTCP protocol, latency-controlled end-to-end aggregation protocol, LEAP for short. The LEAP schedules/distributes packets over multiple parallel links based on user-defined QoS constraints.

Zhang et al. [57] proposed ReLeS, a Reinforcement Learning based Scheduler for MPTCP, which applies deep reinforcement learning (DRL) techniques to learn a neural network (NN) to find the best MPTCP distribution policy. The employed reward function is complex as it determines multi QoS features. For performance evaluation, they consider the following metrics: application goodput, application delay, number out-of-order bytes at receiver (indication of HOL-blocking, and download time. ReLeS significantly outperforms the state-of-the-art schedulers in terms of adaptive learning and real-time scheduling. The main concern is the complexity of the optimization function which may increase the optimization time in complex scheduling scenarios. Wu et al. [54] proposed Peekaboo, a novel learning-based multipath QUIC (MPQUIC) scheduler in that keeps monitoring the impact caused by the current dynamicity level of each path and selects the most suitable scheduling strategy accordingly. According to the reward function, paths with highest throughput are selected. The proposed model is a mixed strategy of deterministic and online adaptive models. To reduce learning complexity, Peekaboo only considers available paths, i.e., have enough CWND. Peekaboo consistently offers superior or similar performance to the best state of the art schedulers, with the performance improvements of Peekaboo reaching by up to 31.2% in emulated networks and up to 36.3% in real network scenarios.

Pokhrel and Williamson [46] applied game theory to define a utility function for cooperative MPTCP. The idea is to model MPTCP subflows contests as a game with common coupled constraints, such that, if a subflow fails to achieve the constraints common to other subflows (such as throughput), all other subflows will be affected. Corresponding issues like out-of-order packet delivery of packet lost are common for cooperative MPTCP subflows. Network emulation experiments show higher throughput and responsiveness in compare to baseline techniques for handling heterogeneous network paths. Huang et al. [20] proposed a distributed Deep Reinforcement Learning (DRL)-based congestion control algorithm in MPTCP. The technique can effectively avoid the HoL-blocking with BLEST scheduler under complex network condition and improve the throughput by 11% compared to the previous MPTCP congestion control algorithms. The C-TCP applies a mixed congestion control strategy to maintain loss and delay in congestion windows [52]. Pokhrel and Garg [38] designed a deep Q-learning based framework for scheduling traffic in IoT applications using deep Qnetwork [29] which is a common benchmark to build an evaluate applications in deep reinforcement domain. A

deep Q-network is designed to achieve human-like intelligence through learning policies directly from high-dimensional data using end-to-end reinforcement learning. Abbasloo et al. [1] proposed DeepCC which leverages advanced deep reinforcement learning (DRL) techniques to let machines automatically learn how to steer throughput-oriented TCP algorithms toward achieving applications' desired delays in a highly dynamic network such as the cellular network. DeepCC fundamentally differs from current learning-based schemes as it attempts to use learning-based techniques to help and boost the performance of the existing TCP schemes instead of replacing them. DeepCC's aim to keep the average delay of packets below applications' desired Targets while keeping the throughput high. Even though MPTCP offers reliable and high-performance connection, but, privacy still a major concern when sending data over multi network interfaces for continuous connectivity. Pokhrel and Choi [33] proposed a blockchain-based federated learning (BFL) design to achieve high communication/data privacy and reduce delay for distributed on-vehicle machine learning processing and data exchange. Analytical results show model efficiency to reduce system delay by utilizing communication an adaptive arrival rate blocking based on current network conditions. The work extended to decentralized fashion to reduce the implications of centralized control on the blockchain mechanism [35]. Later, the authors proposed a federated learning framework to preserve IoV privacy and improve the IoV communication performance [36]. To meet the challenges of IoT Big Data, in terms of variety and volume, the learning process of TCP protocols can be accelerated [17].

Paper	Technique	Objective
Lim et al. [26]	Heuristics	Earliest Completion First
Pokhrel et al. [34]	Heuristics	throughput-based
Chung et al. [10]	ML predictive	QoS
Beig et al. [7]	Heuristics	throughput-based
Pokhrel and Mandjes [40]	Q-Learning	QoS
Chiariotti et al. [9]	Heuristics	QoS
Zhang et al. [57]	RL	QoS
Wu et al. [54]	RL	highest throughput
Pokhrel and Williamson [46]	game theory	throughput and responsiveness
Huang et al. [20]	A distributed (DRL)	loss and delay
Pokhrel and Garg [38]	DRL	QoS
Abbasloo et al. [1]	DRL	delays
Pokhrel and Choi [33]	Federated learning (BFL)	privacy-based

Table 1: provides general comparison between some of heuristics and learning-based MPTCP scheduling techniques.

2.3 Problem statement and research gap

IoT embraces the emergence of new forms of network applications and technologies, which moves towards more complex structures and high resource demanded in terms of computing and networking. The enormous increase of IoT devices leads to new

generation workload model, called IoT Big Data [8, 3]. IoT Big Data commonly refers to the “3Vs” model, that is, the increase of volume, velocity, and variety of data [13, 5]. However, the current MPTCP techniques provide inflexible transmission strategies to meet the performance requirements of IoT Big Data of high responsiveness and high reliability as they do not consider the huge traffic of data in mostly high speed [55]. For example, Pokhrel et al. [44] studied the application of MPTCP on Industrial IoT, so called IIoT, domain. IIoT is an extension of IoT employed in industry for high operational efficiency and intelligent monitoring and tracking applications, and predictive and preventive maintenance [50]. Even though the literature presents promising MPTCP techniques to handle IoT networks, such as IoV. But features like data size and speed at large scale are not well-studied. Moreover, challenges of heterogeneous networks, like mobile networks, including network reliability and timeliness need to be further investigated to consider in context of IoT devices mobility and low-energy contexts [21].

In this work, we will investigate the MPTCP scheduling in IoT Big data and how jointly optimization of maximizing network goodput and reducing delay. Moreover, we will study the development of DRL technique with federated learning to benefit from sharing experience from different network conditions. Finally, the proposed model will be evaluated in comparison with state of the art MPTCP schedulers.

3 Research Design & Methodology

3.1 Research project description

Today, mobile devices like smartphones are supported with various wireless radio interfaces including cellular (3G/4G/LTE) and Wi-Fi (IEEE 802.11) [46]. The legacy devices can only communicate with only one interface. The Transmission Control Protocol, or TCP, has a limitation inability to change connection settings without breaking the connection. Multi-path TCP (MPTCP) protocol has been proposed to solve TCP single-interface limitation and provides a huge improvement on application performance by using multiple paths transparently (auto path changing). MPTCP extends the TCP by enabling single byte stream split into multiple byte streams and transfer them over multiple disjoint network paths or subflows. A MPTCP connection combines a set of different subflows where each subflow performance depends on the condition of its path (including packet loss rate, queue delay, and throughput capacity). Unreliable packet scheduling may lead to critical networking issues such as the head-of-line (HoL) blocking where the packets scheduled on the low-latency path must wait for the packets on the high-latency path to ensure in-order delivery and the out-of-order (OFO) packets, the receiver must maintain a large queue to reorganize the received packets. In this project, we aim to study and experiment MPTCP scheduling on dynamic networks (like cellular network) and try to propose a MPTCP schema which can be effective to overcome limitations of dynamic networks performance [41].

3.2 Research Methodology

3.2.1 Conceptual framework

In this work, we will extend the DeepCC ([1]) by modifying its components to align the work on MPTCP. DeepCC leverages advanced deep reinforcement learning (DRL) techniques to let machines automatically learn how to steer throughput-oriented TCP algorithms toward achieving applications' desired delays in a highly dynamic network such as the cellular network. Figure 3 presents the DeepCC framework. DeepCC fundamentally differs from current learning-based schemes as it attempts to use learning-based techniques to help and boost the performance of the existing TCP schemes instead of replacing them. The DeepCC system model can be described as:

Figure 3: DeepCC MPTCP Conceptual Framework [1]

- The Monitor block: periodically collects $cwnd$ of the system and required packet statistics from the Kernel. Statistics: the average packets' delay per RTT (d), the number of samples (n) used for calculation of d , the average delivery rate per RTT (p) and $cwnd$: The current $cwnd$ calculated by the underlying TCP.
- State generator block: employs the information collected by Monitor block to generate a proper state vector declaring the state of the environment during that

time period. Used techniques: 1) the filter kernel (high dynamic state change): reduce the input space, 2) the recurrent structure (learn cwnd policy in a highly environment): save heuristics about environment state.

- The Reward Generator: the reward function is the moving average of the recent values of the average packets' delay per RTT (d).
- DRL-agent: considers the generated state vector, the application's given Target, and the reward associated to the current state vector (which is generated by Reward block) to decide the proper maximum value of cwnd (cwndmax) based on the previously learned behavior of the environment.
- The final refined cwnd considering this maximum value will be used to send packets into the network.

3.2.2 Experiment details

The experiment will have the following steps:

1. Simulation environment setup. In this setup, we will prepare and install all required packages on the target VM.
2. Re-producing DeepCC. This step involves re-producing DeepCC experiment results by running the free-access code on GitHub. This gives good opportunity to resolve any technical issues of implementing and running the modified version in future.

Figure 4: QoS Learning-based MPTCP Conceptual Framework

3. Design. Through deep analysis of DeepCC framework and other related frameworks, we will decide on which DeepCC framework should be modified to meet the objective of testing MPTCP scheduling in cellular network. Figure 4 shows the main components of the proposed QoS Learning-based MPTCP Conceptual Framework, which can be illustrated as:

- The target network with availability of multi-paths (sub-flows).
 - Statistics about the network and multi-paths are collected at real-time by the MPTCP logic.
 - The network monitor collects the current network state to be used by the state generator.
 - The state generator feeds the Deep RL model with possible states to be evaluated according to some actions.
 - The Deep RL model is responsible for finding the best MPTCP scheduling policy which determines the best schedule (actions) according to network state based on the maximum reward value, which is in our cases a joint value of min delay and max throughput.
 - Application QoS constraints are used by the performance evaluator to evaluate a scheduling plan.
4. Implementation. Based on previous step, required components will be implemented and tested (unit testing).
 5. System training. This step involves training the DRL models for predicting the best MPTCP setup based on current sub-flows status.
 6. System evaluation. The model will be evaluated based on the performance of maximizing the network goodputs while meeting delay and loss rate constraints. Part of input data will be used for evaluation.

The performance of the MPTCP scheduling will be measured based on meeting application delay constraint and improving overall network goodputs over network multi-paths or subflows.

3.3 Data description and data collection

To evaluating the MPTCP performance of DeepCC, we used MPTCP data traces provided by De Coninck et al. [11], which are traces from real smartphone users Contributed. The dataset covers the traffic generated by 12 users using Nexus 5 smartphones running Android 4.4 with a modified Linux kernel that includes Multipath TCP v0.89.5. The authors analysed a trace from a SOCKS proxy serving smartphones using Multipath TCP. The analysis provided on their paper "A first analysis of multipath tcp on smartphones" [12] confirms the influence of heterogeneity of wireless and cellular networks on scheduling of Multipath TCP. The analysis shows that most of the additional subflows are never used to send data. The number of reinjections is also quantified and shows that they are not a major issue for the deployment of Multipath TCP.

3.3.1 Expected outcomes

The expected outcome of this project is an modified version of DeepCC framework which can improve the performance of MPTCP scheduling over cellular networks.

4 Artefact Development Approach

4.1 Deep Reinforcement Learning (DRL) implementation

Figure 5 shows a typical DRL framework, which will be used for the purpose of implementing the proposed framework.

- **Environment:** the MPTCP network. Each path has the following attributes: interface, RTT, loss rate, bandwidth, etc.
- **State:** which represents the current state of all available paths (sub-flows). We will simulate the states based on a given dataset.
- **Reward:** this function evaluates the performance of a MPTCP schedule. The best route is with the reward with the least possible penalties. Moreover, we will experiment different value ranges of gamma (the discount factor). It quantifies how much importance we give for future rewards.
- **Action:** the DNN is responsible for generating actions (MPTCP schedules) to be tested in the environment and then evaluate the reward accordingly. A simple implementation of the DNN will be adopted.

The progress of artefact development can be described as:

- Understanding the Q-learning optimization framework (as described earlier).
- Analysing the main components and the learning behaviour of the DeepCC platform. We consider the DeepCC as a baseline architecture. However, there are core changes we need to implement to meet the work on MPTCP under bi-objective optimization: maximize goodput and minimize lose while meeting application deadline.

Figure 5: Reinforcement Learning with policy represented via DNN [28]

5.1 Research Questions (RQs)

The two main research questions of this thesis are:

1. What are the main challenges of MPTCP Schedulers over future wireless last mile networks?
2. How can experience driven approach enhance the performance of MPTCP Schedulers over future networks?

The enormous increase in number of connected smart-devices (including smart-phones and IoT) embraces huge generation of mobile Internet traffic. The last mile is the final networking segment which carried all network traffic. The available bandwidth in last-mile link can be effectively harms the network throughput as it limits the amount of transmitted data. Another issue is the single point of failure which might cause whole network failure. The quality of the last mile networks significantly determines the reliability and quality of the carrying network. The complex and often inefficient route for packet transmission must take before finally reaching their destination is known as the last mile problem. Although a network can have high average transfer speeds, data will need to hop, skip, and jump along multiple different connections before reaching its destination. In most cases, these connections will have lower bandwidth and involve routers with lower goodputs, which can significantly reduce overall data transfer speeds.

MPTCP can provide a convenient solution for the last mile problem. A MPTCP scheduler needs to provide significant packet routing schedules based on the current status of paths (sub-flows) in terms of loss rate, bandwidth and jitter, in a way, maximizing the network goodput. However, current heuristics-based scheduling are not sufficient to efficient goodput-based solutions particularly for dynamic networks as network conditions are not predictable. One solution is to propose an adaptive scheduling techniques which evolve continues learning from scheduling experiences in attaining a good predictive model. The idea here is to capture the behaviour of last-mile networks to decide on best packet scheduling over available sub-flows considering overall network performance as well as load balancing / workload distribution while meeting QoS measurements, which can be application-defined such as maximum delay and minimum throughput.

6 Results & Discussion

The following are the results of using DeepCC [1] on five MPTCP datasets, which are collected by the the same users (same mobile devices) at different capture time [11].

6.1 Dataset 1: mptcp-dump-20150308-21403700.pcap

- Average capacity: 37.35 Mbits/s
- Average throughput: 2.49 Mbits/s (6.7% utilization)
- 95th percentile per-packet queuing delay: 66 ms
- Average per packet delay: 191 ms

Figure 7: Results for dataset 1

6.2 Dataset 2: mptcp-dump-20150308-21403713.pcap

Figure 8: Results for dataset 2

- Average capacity: 42.18 Mbits/s
- Average throughput: 2.35 Mbits/s (5.6% utilization)
- 95th percentile per-packet queuing delay: 97 ms
- Average per packet delay: 332 ms

Figure 9: Results for dataset 3

6.3 Dataset 3: mptcp-dump-20150408-14121312.pcap

- Average capacity: 78.99 Mbits/s
- Average throughput: 8.37 Mbits/s (10.6% utilization)
- 95th percentile per-packet queuing delay: 30 ms
- Average per packet delay: 112 ms

6.4 Dataset 4: mptcp-dump-20150408-14121306.pcap

Figure 10: Results for dataset 4

- Average capacity: 37.15 Mbits/s
- Average throughput: 4.99 Mbits/s (13.4% utilization)
- 95th percentile per-packet queuing delay: 44 ms
- Average per packet delay: 164 ms

6.5 Dataset 5: mptcp-dump-20150408-14121315

Figure 11: Results for dataset 5

- Average capacity: 79.46 Mbits/s
- Average throughput: 5.54 Mbits/s (7.0% utilization)
- 95th percentile per-packet queuing delay: 40 ms
- Average per packet delay: 142 ms

Some general observations about the results

- Very low MPTCP utilization and subflows usage for all datasets. This denotes that some subflows are not sufficiently used due to unstable network performance. With the establishment of unused subflows. This opens new areas of improvements to adapt Multipath TCP with the smartphone case, in particular the path manager [12].
- There is a negative correlation between communication length (duration) and MPTCP throughput. MPTCP throughput (capacity) decreased as communication time is going. This provides high a high indication on MPTCP performance to network heterogeneity as the the dataset represents data captured by mobile devices at different communication conditions.

- For Dataset 3, which has the small queuing latency and overall packet delivery latency, 30 ms and 112 ms, respectively, the capacity, throughput and utilization are the highest among all Datasets with 79 Mbits/s, 8.37 Mbits/s and 10.6%, respectively.

7 Conclusion & Future Work

The Multi-path TCP (MPTCP) protocol has been proposed to solve TCP single-interface limitation and provides a huge improvement on application performance by using multiple paths transparently (auto path changing). For example, the quality of the last mile networks significantly determines the reliability and quality of the carrying network. MPTCP can provide a convenient solution for the last mile problem. MPTCP scheduling is an adaptive process of determining the best packet distribution over available subflows. This work discuss various scheduling approaches such as heuristic-based and learning-based. Due to the heterogeneity challenge of current computer networks, the learning-based techniques show good MPTCP performance prediction (in terms of goodput and capacity) with the variation on network parameters.

In this work, we provided an experiment to evaluate the performance of MPTCP on heterogeneous network (mobile network) using an existing learning-based MPTCP scheduling framework. The main findings are 1) the poor usage of MPTCP subflows can lead to low network utilization and throughput, 2) communication stability has a high contribution on MPTCP performance and 3) decreasing queuing latency and overall packet delivery latency can significantly improve MPTCP throughput and utilization.

For future work, we are planning to implement the proposed learning-based framework and evaluate it for time-sensitive application under heterogeneous (mobile-based) networks.

References

- [1] S. Abbasloo, C.-Y. Yen, and H. J. Chao, Wanna make your TCP scheme great for cellular networks? let machines do it for you!, *IEEE Journal on Selected Areas in Communications*, 39 (2020), pp. 265–279.
- [2] V. Adarsh, P. Schmitt, and E. Belding, Mptcp performance over heterogenous subpaths, in *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, IEEE, 2019, pp. 1–9.
- [3] E. Ahmed, I. Yaqoob, I. A. T. Hashem, J. Shuja, M. Imran, N. Guizani, and S. T. Bakhsh, Recent advances and challenges in mobile big data, *IEEE Communications Magazine*, 56 (2018), pp. 102–108.
- [4] S. Bae, D. Ban, D. Han, J. Kim, K.-h. Lee, S. Lim, W. Park, and C.-k. Kim, Streetsense: Effect of bus wi-fi aps on pedestrian smartphone, in *Proceedings of the 2015 Internet Measurement Conference*, 2015, pp. 347–353.
- [5] M. Bansal, I. Chana, and S. Clarke, A survey on iot big data: Current status, 13 v’s challenges, and future directions, *ACM Computing Surveys (CSUR)*, 53 (2020), pp. 1–59.
- [6] S. Barré, C. Paasch, and O. Bonaventure, Multipath TCP: from theory to practice, in *International conference on research in networking*, Springer, 2011, pp. 444–457.
- [7] E. F. G. M. Beig, P. Daneshjoo, S. Rezaei, A. A. Movassagh, R. Karimi, and Y. Qin, Mptcp throughput enhancement by q-learning for mobile devices, in *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, pp. 1171–1176.
- [8] M. Chen, Y. Liu, and S. Mao, Big data: A survey mobile network and applications, (2014).
- [9] F. Chiariotti, S. Kucera, A. Zanella, and H. Claussen, Analysis and design of a latency control protocol for multi-path data delivery with pre-defined QoS guarantees, *IEEE/ACM Transactions on Networking*, 27 (2019), pp. 1165–1178.
- [10] J. Chung, D. Han, J. Kim, and C.-k. Kim, Machine learning based path management for mobile devices over mptcp, in *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, IEEE, 2017, pp. 206–209.
- [11] Q. D. Coninck, M. Baerts, B. Hesmans, and O. Bonaventure, CRAWDAD dataset uclouvain/mptcp_smartphone (v. 2016-03-04). Downloaded from https://crawdad.org/uclouvain/mptcp_smartphone/20160304, Mar. 2016.
- [12] Q. De Coninck, M. Baerts, B. Hesmans, and O. Bonaventure, A first analysis of multipath tcp on smartphones, in *International Conference on Passive and Active Network Measurement*, Springer, 2016, pp. 57–69.

- [13] H. Fang, Z. Zhang, C. J. Wang, M. Daneshmand, C. Wang, and H. Wang, A survey of big data research, *IEEE network*, 29 (2015), pp. 6–9.
- [14] S. Ferlin, Ö. Alay, O. Mehani, and R. Boreli, Blest: Blocking estimation-based MPTCP scheduler for heterogeneous networks, in 2016 IFIP Networking Conference (IFIP Networking) and Workshops, IEEE, 2016, pp. 431–439.
- [15] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, and C. Paasch, Rfc 6824: TCP extensions for multipath operation with multiple addresses, Internet Engineering Task Force, (2013).
- [16] Y. E. Guo, A. Nikraves, Z. M. Mao, F. Qian, and S. Sen, Accelerating multipath transport through balanced subflow completion, in Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking, 2017, pp. 141–153.
- [17] S. Ha, I. Rhee, and L. Xu, Cubic: a new TCP-friendly high-speed TCP variant, *ACM SIGOPS operating systems review*, 42 (2008), pp. 64–74.
- [18] H. Han, S. Shakkottai, C. Holot, R. Srikant, and D. Towsley, Overlay TCP for multi-path routing and congestion control, in IMA Workshop on Measurements and Modeling of the Internet, 2004.
- [19] M. Honda, Y. Nishida, L. Eggert, P. Sarolahti, and H. Tokuda, Multipath congestion control for shared bottleneck, in Proc. PFLDNeT workshop, vol. 357, Citeseer, 2009, p. 378.
- [20] C. Huang, J. Zhang, and T. Huang, Objective-oriented resource pooling in MPTCP: A deep reinforcement learning approach, in 2020 3rd International Conference on Hot Information-Centric Networking (HotICN), IEEE, 2020, pp. 175–181.
- [21] H. Karim, IoT networking using mptcp protocol, 2020.
- [22] F. Kelly and T. Voice, Stability of end-to-end algorithms for joint routing and rate control, *ACM SIGCOMM Computer Communication Review*, 35 (2005), pp. 5–12.
- [23] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, and J.-Y. Le Boudec, MPTCP is not pareto-optimal: performance issues and a possible solution, in Proceedings of the 8th international conference on Emerging networking experiments and technologies, 2012, pp. 1–12.
- [24] N. Kuhn, E. Lochin, A. Mifdaoui, G. Sarwar, O. Mehani, and R. Boreli, Daps: Intelligent delay-aware packet scheduling for multipath transport, in 2014 IEEE International Conference on Communications (ICC), IEEE, 2014, pp. 1222–1227.
- [25] Y.-s. Lim, Y.-C. Chen, E. M. Nahum, D. Towsley, and K.-W. Lee, Cross-layer path management in multi-path transport protocol for mobile devices, in IEEE INFOCOM 2014-IEEE Conference on Computer Communications, IEEE, 2014, pp. 1815–1823.

- [26] Y.-s. Lim, E. M. Nahum, D. Towsley, and R. J. Gibbens, Ecf: An MPTCP path scheduler to manage heterogeneous paths, in Proceedings of the 13th international conference on emerging networking experiments and technologies, 2017, pp. 147–159.
- [27] Y. Liu, A. Neri, A. Ruggeri, and A. M. Vegni, A MPTCP-based network architecture for intelligent train control and traffic management operations, *IEEE Transactions on Intelligent Transportation Systems*, 18 (2016), pp. 2290–2302.
- [28] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, Resource management with deep reinforcement learning, in Proceedings of the 15th ACM workshop on hot topics in networks, 2016, pp. 50–56.
- [29] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, *nature*, 518 (2015), pp. 529–533.
- [30] S. C. Nguyen and T. M. T. Nguyen, Evaluation of multipath TCP load sharing with coupled congestion control option in heterogeneous networks, in Global Information Infrastructure Symposium-GIIS 2011, IEEE, 2011, pp. 1–5.
- [31] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, Experimental evaluation of multipath TCP schedulers, in Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop, 2014, pp. 27–32.
- [32] Q. Peng, A. Walid, J. Hwang, and S. H. Low, Multipath TCP: Analysis, design, and implementation, *IEEE/ACM Transactions on networking*, 24 (2014), pp. 596–609.
- [33] S. R. Pokhrel, Federated learning meets blockchain at 6g edge: A drone-assisted networking for disaster response, in Proceedings of the 2nd ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond, 2020, pp. 49–54.
- [34] S. R. Pokhrel and J. Choi, Low-delay scheduling for internet of vehicles: Load-balanced multipath communication with FEC, *IEEE Transactions on Communications*, 67 (2019), pp. 8489–8501.
- [35] —, A decentralized federated learning approach for connected autonomous vehicles, in 2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), IEEE, 2020, pp. 1–6.
- [36] —, Improving TCP performance over wifi for internet of vehicles: A federated learning approach, *IEEE Transactions on Vehicular Technology*, 69 (2020), pp. 6798–6802.
- [37] S. R. Pokhrel, J. Ding, J. Park, O.-S. Park, and J. Choi, Towards enabling critical mmhc: A review of urllc within mmhc, *IEEE Access*, 8 (2020), pp. 131796–131813.
- [38] S. R. Pokhrel and S. Garg, Multipath communication with deep q-network for industry 4.0 automation and orchestration, *IEEE Transactions on Industrial Informatics*, (2020).

- [39] S. R. Pokhrel, J. Jin, and H. Le Vu, Mobility-aware multipath communication for unmanned aerial surveillance systems, *IEEE Transactions on Vehicular Technology*, 68 (2019), pp. 6088–6098.
- [40] S. R. Pokhrel and M. Mandjes, Improving multipath TCP performance over wifi and cellular networks: An analytical approach, *IEEE Transactions on Mobile Computing*, 18 (2018), pp. 2562–2576.
- [41] S. R. Pokhrel, L. Pan, N. Kumar, R. Doss, and H. Le Vu, Multipath TCP meets transfer learning: A novel edge-based learning for industrial IoT, *IEEE Internet of Things Journal*, (2021).
- [42] S. R. Pokhrel, M. Panda, and H. L. Vu, Fair coexistence of regular and multipath TCP over wireless last-miles, *IEEE Transactions on Mobile Computing*, 18 (2018), pp. 574–587.
- [43] S. R. Pokhrel, R. C. Pandey, and S. R. Joshi, Performance modelling and evaluation of V2I video surveillance system, in *2015 Twelfth International Conference on Wireless and Optical Communications Networks (WOCN)*, IEEE, 2015, pp. 1–4.
- [44] S. R. Pokhrel, Y. Qu, and L. Gao, Qos-aware personalized privacy with multipath tcp for industrial iot: Analysis and design, *IEEE Internet of Things Journal*, 7 (2020), pp. 4849–4861.
- [45] S. R. Pokhrel and S. Singh, Compound TCP performance for industry 4.0 wifi: A cognitive federated learning approach, *IEEE Transactions on Industrial Informatics*, 17 (2020), pp. 2143–2151.
- [46] S. R. Pokhrel and C. Williamson, A rent-seeking framework for multipath TCP, (2020).
- [47] J. Postel, Rfc0793: Transmission control protocol, 1981.
- [48] C. Raiciu, M. Handley, and D. Wischik, Coupled congestion control for multipath transport protocols, tech. rep., IETF RFC 6356, Oct, 2011.
- [49] M. Scharf and S. Kiesel, Nxg03-5: Head-of-line blocking in TCP and SCTP: Analysis and measurements, in *IEEE Globecom 2006*, IEEE, 2006, pp. 1–5.
- [50] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, Industrial internet of things: Challenges, opportunities, and directions, *IEEE Transactions on Industrial Informatics*, 14 (2018), pp. 4724–4734.
- [51] J. Sommers and P. Barford, Cell vs. wifi: on the performance of metro area mobile connections, in *Proceedings of the 2012 internet measurement conference*, 2012, pp. 301–314.
- [52] K. Tan, J. Song, Q. Zhang, and M. Sridharan, A compound TCP approach for high-speed and long distance networks, in *Proceedings-IEEE INFOCOM*, 2006.

- [53] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, Design, implementation and evaluation of congestion control for multipath TCP., in NSDI, vol. 11, 2011, pp. 8–8.
- [54] H. Wu, Ö. Alay, A. Brunstrom, S. Ferlin, and G. Caso, Peekaboo: Learning-based multipath scheduling for dynamic heterogeneous environments, *IEEE Journal on Selected Areas in Communications*, 38 (2020), pp. 2295–2310.
- [55] Y. Xing, J. Han, K. Xue, J. Liu, M. Pan, and P. Hong, MPTCP meets big data: Customizing transmission strategy for various data flows, *IEEE Network*, 34 (2020), pp. 35–41.
- [56] C. Yu, W. Quan, N. Cheng, S. Chen, and H. Zhang, Coupled or uncoupled? multi-path TCP congestion control for high-speed railway networks, in 2019 IEEE/CIC International Conference on Communications in China (ICCC), IEEE, 2019, pp. 612–617.
- [57] H. Zhang, W. Li, S. Gao, X. Wang, and B. Ye, Reles: A neural adaptive multipath scheduler based on deep reinforcement learning, in IEEE INFOCOM 2019-IEEE Conference on Computer Communications, IEEE, 2019, pp. 1648–1656.