On the Performance of TCP in Reconfigurable Data Center Networks

Kaan Aykurt*, Johannes Zerwas*, Andreas Blenk*†, Wolfgang Kellerer*
*Chair of Communication Networks, Technical University of Munich, Germany
†Siemens AG, Munich, Germany

***Abstract*—Today's data centers are hosting various applications under the same roof. The diversity among deployed applications leads to a complex traffic mix in Data Center Networks (DCNs). Reconfigurable Data Center Networks (RDCNs) have been designed to fulfill the demanding requirements of ever-changing data center traffic. However, they pose new challenges for network traffic engineering, e.g., interference between reconfigurations and congestion control (CC). This raises a fundamental research problem: can the current transport layer protocols handle frequent network updates?**

This paper focuses on the Transmission Control Protocol (TCP) and presents a measurement study of TCP variants in RDCNs. The quantitative analysis of the measurements shows that migrated flows suffer from frequent reconfigurations. The effect of reconfigurations on the cost, e.g. increased Flow Completion Time (FCT), depending on the traffic mix is modeled with Machine Learning (ML) methods. The availability of such a model will provide insights into the relationship between the reconfiguration settings and the FCT. Our model explains 88% of the variance in the FCT increase under different reconfiguration settings.

***Index Terms*—reconfigurable data center networks, TCP measurements**

I. Introduction

Over the past decades, the introduction of cloud computing transformed a significant portion of the information technologies industry [1]. With this transformation, many new applications appeared in DCNs with different requirements. For instance, distributed machine learning applications demand high bandwidth to transmit large amounts of data while other applications need to fulfill strict latency and high availability constraints such as deployments of 6G core networks [2].

As a result, the bandwidth and latency requirements of modern data center networks are far more different than the conventional infrastructures. Conventional fully provisioned static DCNs are either too costly or fail to meet this challenging demand since it is not feasible to create permanent high bandwidth links across the racks [3]. Therefore, the literature mainly focuses on meeting the challenging demands of modern network architectures by introducing reconfigurable network elements, such as Optical Circuit Switches (OCS), that enable temporary high bandwidth connections between source and destination pairs on demand [3]–[7]. The creation and destruction of new links cause the flows to be re-routed to different links depending on the reconfiguration scheme of the OCS. This process causes the flows to be migrated without the knowledge of the receiver or sender.

The RDCNs enable better performance for DCNs by establishing low latency and high throughput. However, they also present a research question: what is the effect of introducing temporary paths with different provisioning periods on the FCT performance?

The performance analysis of RDCNs exists in the literature to some extent [3], [5]–[8]. However, these analyses mainly take into account the flow-level considerations [3], [5]–[7] or take a theoretical analysis perspective [5], [8]. In reality, consideration of packet-level characteristics may pose a challenge to extract the full potential of RDCNs. Combinations of high-bandwidth circuit networks in addition to the traditional packet-switched networks introduce non-trivial problems such as flow interruptions due to reconfiguration downtimes and bandwidth fluctuations [9]. Consequently, the changes in the bottleneck link capacity that emerges from path reconfigurations pose a problem for end-to-end network connections: can transport layer protocols utilize the link capacity efficiently with rapid fluctuations in the available bandwidth?

The heterogeneity of network traffic loads, the dynamic nature of modern data centers, and the existence of complex, diverse paths between two end-hosts make the role of network transport protocols more critical for ensuring seamless end-to-end connectivity. TCP is the current *de facto* standard transport protocol of modern DCN architectures. TCP connections adjust their sending rates according to the available bandwidth. The rate limitations, in general, are modeled with the consideration of the capacity of a bottleneck link and a particular Round Trip Time (RTT). The networking community has analyzed TCP behavior in static environments [10]–[13]. However, in heterogeneous cloud environments, newly introduced versions of TCP as well as different transport layer protocols coexist [14]. Available studies for TCP performance in dynamic environments are also limited and lack the interaction of multiple flows [9]. To the best of our knowledge, performance analysis of TCP behavior on dynamic architectures with coexisting flows of different natures is not yet available in the literature.

This paper analyzes the FCT performance of various TCP variants under different reconfiguration schemes. To this end, it provides testbed measurements of TCP flows under various reconfiguration settings. The measurement results are used to model the effect of frequent path migrations on the migrated flow. In particular, this model predicts the increase of the FCT given the reconfiguration settings and involved TCP

variants. The outcomes of the model provide insights into the interaction between the reconfiguration scenarios and the FCT.

This work contributes to the networking community by shedding light on the interactions of various TCP variants in a dynamic environment. The findings of this paper indicate that both the traffic mix and the reconfiguration period in a dynamic environment affect TCP performance.

The remainder is structured as follows: Sec. II gives a brief overview of RDCN and widely used transport layer protocols. Sec. III lists related measurement studies. Sec. IV describes the testbed and measurement procedure. Sec. V presents analysis of flow migrations under different reconfiguration scenarios. Finally, Sec. VI introduces an ML model to predict the FCT prolongation. We conclude and discuss future work in Sec VII.

II. Background

This section gives an introduction to RDCNs and briefly provides an overview of widely utilized TCP variants.

A. Reconfigurable Data Center Networks

Fig. 1 shows a hybrid RDCN with N racks. The traditional, static packet-switched network is augmented by a circuit switching element to provide rack-level direct connectivity. The packet network is a static environment, where the topology cannot be configured. The circuit switch, typically realized by a reconfigurable device such as an OCS, introduces dynamicity to an RDCN. In a typical RDCN, all racks are connected via the packet network permanently. Additionally, the circuit switch exists to create temporary links between rack pairs on demand. Limited availability on ports of the circuit switch and the high financial costs of full provisioning means that the high bandwidth network cannot be utilized all the time by the racks. Therefore, a scheduling algorithm for the circuit switch network is required to connect rack pairs on demand.

The authors of [15], [16] showed that during circuit reconfiguration, no circuit links can be used. In RDCNs, a typical circuit reconfiguration schedule consists of 90% circuit up-times and a 10% circuit downtime [17]. We refer to circuit up-time as the *day time* and downtime as the *night time*.

Flow-level simulation studies of RDCNs have shown that their performance is superior in comparison to the static topologies, e.g., [3], [4]. However, these analyses consider 100% link utilization immediately after reconfigurations. When packet-level traffic characteristics are taken into account, the assumption of efficient link usage may not hold. Therefore, the real cost of frequent reconfigurations on RDCN performance requires the analysis of transport layer protocols and their behavior under frequent reconfigurations.

B. TCP Variants

The TCP/IP stack, which is still a part of today's internet, was introduced back in the 1980s [18]. The transport layer presented in the modern TCP/IP stack is implemented in the operating system kernel and has an end-to-end view of the connection, which considers only a single logical link between the two endpoints [19]. However, particular characteristics of individual links in each hop between the endpoints may influence the transport protocol's behavior.

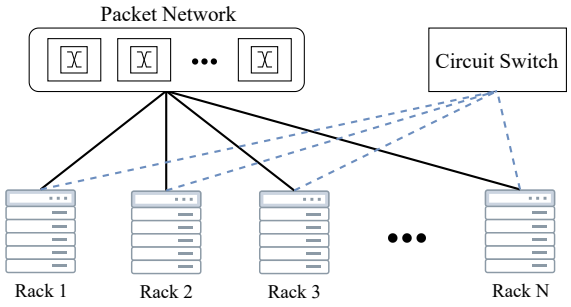


Fig. 1. Overview of an RDCN. A circuit switch is present to introduce dynamicity in addition to the conventional packet network.

TCP is one of the most popular transport layer protocol implementations. It enables as fast as possible message exchange across networking entities with guaranteed delivery and reliability through the acknowledgment mechanism [18]. Although the mechanism appears to be straightforward, there are essential parameters in a TCP connection. One of these parameters is the ordering of the packets received. The burst of packets may change the order in the link and cause overhead to the receiver. These packets have to be reassembled in the receiver. The other important aspect is the size of the group of packets transmitted in each period before waiting for an ACK from the receiver. This is referred to as the bytes-in-flight or the congestion window size in the literature. The determination of this parameter is essential for ensuring fast delivery. A smaller congestion window size leads to slower transmission, whereas a larger congestion window size may cause unreliability. The packet losses incurred by the TCP connection cause retransmissions. Missing packets that are not successfully received by the receiver have to be retransmitted by the TCP protocol to successfully deliver a message stream.

The availability of a set of parameters enables flexibility in design for various TCP implementations. The trade-off between latency and reliability allows for different TCP CC designs. Additionally, these algorithms also determine the distribution of the available bandwidth between flows. Therefore, in a broader sense, the goal of TCP CC is to maximize the sending rate while ensuring reliable and fair communication. Based on their nature, TCP variants can be categorized into five groups: loss-based, delay-based, capacity-based, hybrid, and explicit feedback-based CC.

Loss-based CC: Packet losses are one of the most commonly used congestion signals. The detection of a packet loss triggers the loss-based algorithms to adjust their congestion window size. The investigated loss-based TCP variants in this paper are: CUBIC [20] and Reno [21]. These algorithms grow their congestion window sizes until a packet loss occurs. If a loss occurs after the growth period, this indicates congestion in the link. This causes the congestion window size to shrink. Different implementations use different approaches for increasing

and decreasing the window size. CUBIC estimates the packet loss with a cubic function of the last time since a packet loss occurred. In contrast, Reno employs an Additive Increase, Multiplicative Decrease (AIMD) scheme.

Delay-based CC: Another technique for detecting congestion is continuously monitoring packets' RTT. An increase in RTT is used to indicate queue buildup in the network. Pure RTT-based congestion detection introduces a fairness problem. RTT is the first parameter that increases after more than two flows compete on the same link since the queues build up before a packet loss happens. This introduces a problem for competition for delay-based CC algorithms and causes them to suffer in terms of achieved throughput. This paper omits the analysis of delay-based algorithms due to decreasing popularity.

Capacity-based CC: The links have limited capacity, and it needs to be shared between multiple flows in general. One of the methods of adjusting congestion window size is to estimate the available capacity. The capacity-based algorithms track the estimated bandwidth to adapt the congestion window size after a loss [19]. The estimation phase is determined in the start phase and updated throughout the serving process. An example of this implementation investigated in this paper is TCP Westwood [22].

Hybrid CC: One other alternative to detect congestion is to combine more than one metric. TCP BBR [23] is an example of the category of hybrid CC mechanism which continuously measures RTT and link capacity to determine the congestion window size. The capacity estimation relies on predicting the Bandwidth-Delay Product (BDP). BDP is the product of a link's capacity and the RTT. The consideration of these two parameters enables a hybrid mechanism to detect congestion.

Explicit Feedback-based CC: Finally, a subset of TCP variants relies explicitly on feedback from the network to detect congestion. While TCP is implemented in the operating system kernel, the availability of an external notification field that indicates congestion helps detection much easier and more efficiently. DCTCP [24], PowerTCP [25] and XCP [26] are manifestations of explicit feedback-based CC category. Analysis has shown that explicit feedback can benefit congestion window size adjustment significantly [27]. However, it requires the network elements to support additional Ethernet fields, which are currently not standardized. This paper excludes the analysis of explicit feedback-based CC.

III. Related Work

We are not aware of any thorough analysis of TCP flows subject to frequent migrations. However, several related works exist, which investigate TCP behavior under different scenarios. Most of the existing TCP analyses focus on static DCN topologies and employ simulation studies. The analysis of widely used TCP variants in static DCNs is presented in [27]. Improvements to DCTCP and an overview of existing CC implementations are discussed in [28]. Jain et al. discuss the TCP implementations focusing on wireless and, in general, low-bandwidth lossy links [29]. TCP performance in static DCNs under different CC schemes is analyzed in [30]. Overall, these papers do not take into account the dynamic nature of DCNs.

In RDCNs, the effect of packet re-ordering on TCP throughput has also been analyzed and known for a very long period of time [31]. Recently, Cârpa et al. have built on top of this work and analyzed the effect of switch reconfigurations on TCP performance [32]. This work considers multiple flows competing on the same bottleneck. However, it only focuses on TCP CUBIC and lacks a variation of CC algorithms. Mukerjee et al. analyze TCP performance and suggests an adaptation of TCP based on an open-source RDCN emulator [9]. However, this work lacks the interaction of different TCP variants and does not consider homogeneous links. Existing works focus mainly on simulation studies and do not reflect real-world network paths' complex nature. A testbed implementation for an extensive measurement setup is limited. Moreover, no behavior modeling for TCP in RDCNs is existent.

The successful modeling of TCP throughput may benefit the CC algorithms. Prophet [33] framework provides a model to predict the throughput of TCP flows under static DCNs and strict conditions. Authors of [34] build a machine learning model to infer TCP characteristics from a set of passive measurements in static DCNs. CCAC [35] tool relies on theoretical analysis and presents a model to verify certain properties of CC algorithms before deployment and lacks modeling of multiple flows competing on the same bottleneck. The existing machine learning models predict the throughput to enable better congestion window size adjustment in static DCNs. However, these analyses have a limitation to their approach. The generalized prediction of these models to RDCNs does not hold. In order to close this gap, this paper proposes an ML model to relate the TCP variants and reconfiguration settings to the FCT.

IV. Measurement Setup

The goal of this paper is to compare the network utilization of flows under varying reconfiguration scenarios on a small representative setup. This section establishes a programmable data plane that emulates an OCS and describes the measurement settings.

A. Testbed

Fig. 2 shows the testbed. It consists of three servers, a `Switch` and a `Controller`. `Server 1` and `Server 2` are connected to the `Switch` via two 10 Gbps optical cables. The existence of two separate physical links allows packet generation of up to 20 Gbps in `Server 1` without facing a bottleneck in the uplink to the `Switch`. The server CPU and bus are capable of handling speeds up to 40 Gbps, and hence 20 Gbps traffic generation can be achieved. Four ports of the `Switch` are connected via loopback cables. The objective of these links is to introduce congestion of packets only in the loopback link and hence emulating a more extensive data center architecture where the congestion may occur between the endpoints. The `Measurement Server` is connected to the optical taps in the loopback links. The `Controller`

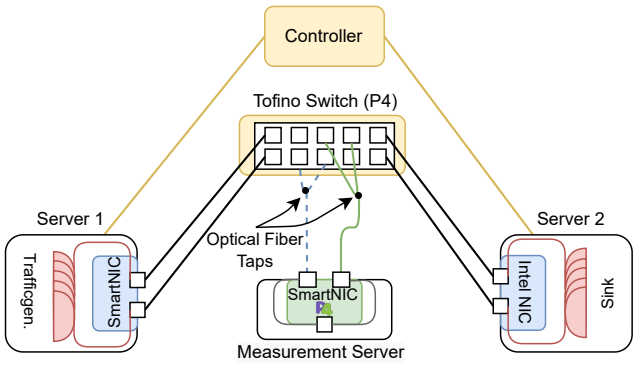


Fig. 2. Testbed overview.

TABLE I
SUMMARY OF THE INVESTIGATED SCENARIOS.

	Values
TCP Variants	CUBIC, BBR, Reno, Westwood
Reconfiguration Periods	None, 1 ms, 5 ms, 10 ms 25 ms, 50 ms, 100 ms
Reconfiguration Downtimes	None, 1 ms, 2 ms, 5 ms, 10 ms
Number of Flows	1, 2, 10, 20
Flow Sizes	2^n MB for n in [0, 8], 2 GB, 4 GB

has 1 Gbps connection with all the entities to orchestrate the experiment.

The traffic is generated by using iPerf version 2.0.10 [36] for flows greater than 1 GB in `Server 1`. Smaller flows are generated via conventional server-client fashion HTTP file transfer. As for the NIC, Netronome Agilio SmartNIC NFP-4000 [37] is used without any custom P4 programs. `Server 2` is designated as the traffic sink, and two separate server processes are being run on two different ports of the Intel X710 series NIC [38].

The incoming packets from `Server 1` to the `Switch` are initially configured to traverse the color-coded loopback links on the `Switch`. The forwarding rules are installed manually. The outgoing direction of the loopback links is tapped, and the data packets are collected in the `Measurement Server`. The `Switch` is then programmed to forward data packets coming from the loopback links to their final destination on `Server 2`. With this architecture, the traffic is tapped passively and the state of TCP is preserved on separate physical ports of `Server 2`.

The `Measurement Server` features a SmartNIC with two ports. A custom P4 program is loaded to SmartNIC to forward incoming packets from physical ports to a virtual interface. During the forwarding process, the IP ToS field of the packets is modified to distinguish the incoming packets from different physical interfaces, as it is not possible to differentiate from the virtual interface otherwise. *TCPDUMP* traffic collection is run on the virtual interface, and all the data packets are collected with software timestamping.

The `Controller` is present to orchestrate the measurement process. In addition to the remote `Controller` shown in Fig 2, the `Switch` has its own local C++ manager, which allows higher time precision for reconfiguration periods. The local manager of the `Switch` is instantiated with the remote `Controller` at the beginning of the experiment.

B. Measurement Procedure

Unless stated otherwise, the flows are created in the `Server 1` at the same time. In the initial state, packets arriving at the `Switch` are forwarded to green and blue (dashed) loopback links such that they follow completely separated paths in the loopback link. In the second state (after one reconfiguration), blue packets traversing the blue loopback link are migrated to the green loopback link, causing congestion. The rules differentiating packets leaving the `Switch` to the destination server are kept intact during the rule update. With this approach, the migration takes place only in the loopback links independent of flow source and destination. This reconfiguration scheme is then applied back and forth with a pre-defined reconfiguration period.

ARP entries to `Server 1` and `Server 2` are installed prior to the measurement. Therefore no ARP broadcasting and messages are being forwarded via the `Switch`. Moreover, the `Switch` is using a single buffer for all the incoming packets from all the ports and hence making the likelihood of ACK packet drops very high. Therefore, unlike the data packets in the outgoing direction, TCP ACK packets are circulated through the `Switch`, meaning that they do not trace the loopback link. By bypassing the loopback link, ACK packets are transported instantly, hence reducing the likelihood of ACK packet drops. This behavior emulates the asymmetric routing in RDCNs, where large flows are mainly subject to migration and small flows are served via the static topology [4].

The summary of analyzed scenarios, parametrized by the TCP variant, reconfiguration period, reconfiguration downtime, number of flows, and flow sizes are presented in Table I. The values column shows the available settings and the complete list of scenarios includes the combination of the specified settings. The most popular TCP variants from different congestion control categories are selected for investigation. Reconfiguration downtimes reflect the widely proposed values in the literature and reconfiguration periods are configured to result in a 90% duty cycle which is the common choice in the literature [3]–[8]. Flow sizes are selected to cover elephant (2 GB and 4 GB) and mice flows. The number of flows represents anticipated edge cases: a single flow, 2 competing flows as well as multiple flows.

In the remainder of this paper, Sec. V presents the most important findings, whereas Sec. VI includes all the measurement results for modeling. The effect of reconfiguration downtime is only evaluated in Sec. V-D and the rest of the analysis focuses on the zero downtime case.

V. Evaluation

Frequent migrations of the flows are expected to affect the achieved throughput of the flows and hence the FCT. The intuition behind this expectation stems from the fact that the flows might take some time to ramp up their sending rates (due to TCP CC algorithm behavior) after the reconfigurations and lead to the under-utilization of the links. In this section, we analyze the packet traces collected during the measurement process to investigate the effect of the reconfiguration period on TCP performance. The results report averages and 95% confidence intervals from 30 measurement runs.

A. Benchmark of the Testbed

In order to establish a benchmark of the testbed for comparing flow behavior in case of congestion, a single TCP flow with 4 GB volume is considered. This flow is migrated between two links with a fixed reconfiguration period for each measurement scenario. The measurement scenarios for benchmarking the testbed include reconfiguration periods of 10 ms, 50 ms, and no migration at all.

Fig. 3 shows the achieved throughput for each TCP variant with respect to time and reconfiguration period. The 95% confidence intervals are also plotted with less opacity in the background with matching colors. However, it is not visible since the variance between the distinct measurement runs is very low. The results indicate that all the TCP variants achieve maximum theoretical throughput in less than 10 ms. Regardless of the reconfiguration period, the flows can achieve 10 Gbps throughput and the flows are complete in 3.47 s as expected by theoretical calculations (also considering the headers in addition to the payload). Since the throughput is not affected by the reconfiguration period, the benchmarks indicate that the `Switch` does not drop packets during reconfiguration and in the absence of congestion frequent path migrations do not affect the FCT.

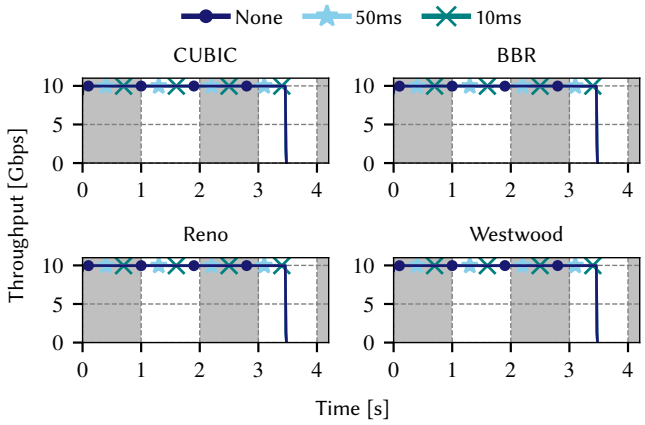


Fig. 3. Average throughput comparison of a single TCP flow under frequent reconfigurations. In the absence of congestion, there is no effect on the FCT.

B. Competition of Two Flows: Same TCP Variant

Congestion is likely to occur in a DCN environment, where multiple flows coexist. Accordingly, we investigate a scenario with two flows of 2 GB each. One of these flows is referred to as the *migrated* flow, which is periodically migrated in the loopback link shown in Fig. 2. The other flow (*main* flow) is served on the same link, which is shown by the green link in Fig. 2. Our migration approach, as introduced in Sec. IV-B, introduces periodic congestion in the loopback link, and after re-migration, the congestion vanishes.

Throughput: Fig. 4 shows the average throughput of the main and migrated flow with TCP CUBIC under different reconfiguration periods. With 50 ms reconfiguration period, the main and migrated flow's FCTs do not differ significantly. A jigsaw pattern of the throughput between 10 Gbps and 5 Gbps is evident. This indicates that during the congestion period the rate is allocated fairly and 50 ms gives enough time for TCP CUBIC to ramp up its sending rate on the uncongested link. However, at 25 ms, a decrease in the average throughput of the migrated flow is observed. This directly translates into an increase in the migrated flow's FCT. Below 10 ms, the magnitude of the FCT increase grows and the 95% confidence intervals of the throughput do not overlap anymore. At 1 ms, the growth of the TCP congestion window after reconfiguration is not fast enough and, hence the migrated flow achieves less throughput consistently. The findings indicate that the FCT prolongation and the reconfiguration period are inversely related, which confirms the observations of Mukerjee et al. [9]. Although not shown in this figure, an increased FCT for the migrated flow is similarly observed for the other TCP variants as well. These extensive measurements serve as a baseline for network operators to fine-tune the reconfiguration period for fair bandwidth allocation.

Congestion Window Size: In order to elaborate on the analysis of varying FCTs with different reconfiguration scenarios, Fig. 5 shows the congestion window size of both flows. A baseline (from the measurements in Sec. V-A) of the congestion window size is also plotted to outline the benchmark values. The ideal CC behavior is to ramp up the maximum size and keep it constant throughout the transmission to achieve a small FCT. For our specific testbed environment, the maximum allowable congestion window size without encountering a packet loss is determined as around 3000 KB from baseline measurements. At 50 ms reconfiguration period, the congestion window size of CUBIC indicates a pattern with peaks and valleys. The valleys correspond to the immediate decrease in the congestion window size after reconfiguration, hence encountering a packet loss due to congestion. The peaks align with the time instances just before reconfiguration, where the algorithm is trying to ramp up its maximum allowable bytes in flight. However, even at 50 ms, the peaks are only at 60% of the baseline values. As the reconfiguration period is decreased further, the jigsaw pattern consisting of peaks and valleys becomes less evident and the migrated flow suffers significantly from small congestion window size. Overall, the analysis of congestion window size yields results parallel to the throughput observations and serves as an explanatory

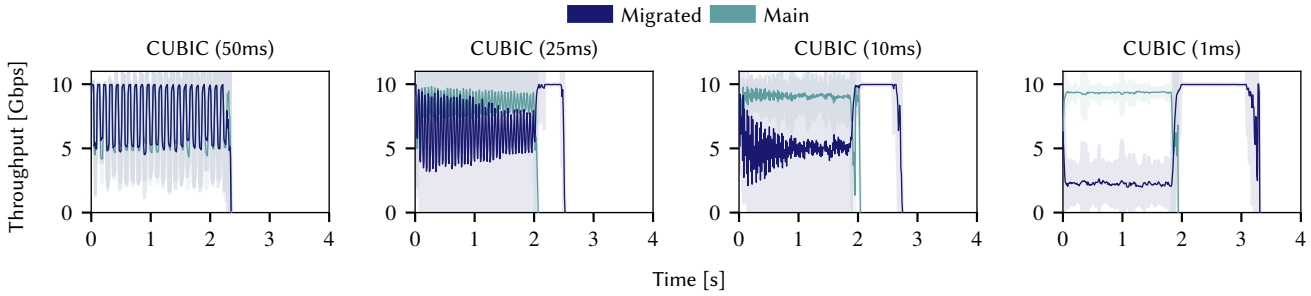


Fig. 4. Average throughput of TCP CUBIC flows with varying reconfiguration periods. 95% confidence intervals are presented in the background with matching colors. The migrated flows ramp up their sending rate after the main flows finish. Overall, decreasing the reconfiguration period increases the migrated flow's FCT.

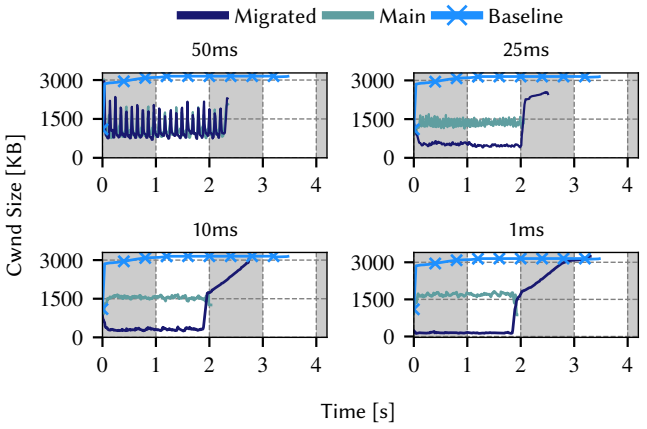


Fig. 5. Average congestion window size of TCP CUBIC under different reconfiguration periods. The congestion introduced by frequent migrations hinders TCP CUBIC's ability to ramp up its congestion window size. The migrated flow can only ramp up its congestion window size after the main flow finishes at around $t = 2\,\mathrm{s}$.

factor for increased FCT of the migrated flow with decreasing reconfiguration period.

C. Competition of Two Flows: Different TCP Variants

Previous studies have shown that many TCP variants are co-existing alongside each other [9], [14]. The goal of comparing two different TCP variants is to shed light on the scenarios involving the interaction of different TCP variants. Fig. 6 shows the average throughput of migrated and main flows of different TCP variants. Similar to the previous sections, it presents the mean throughput and 95% confidence interval per time instance for 30 different measurement runs.

Fig. 6(a) presents the interaction of a migrated flow using CUBIC with main flows that use other variants and a reconfiguration period of $50\,\mathrm{ms}$. Unlike for two competing TCP CUBIC flows, the bandwidth is not shared fairly. The actual bandwidth distribution depends on the traffic mix.

The first sub-figure of Fig. 6(a) shows the main flow using BBR. It can be seen from the figure that although being migrated, CUBIC consistently receives higher throughput than BBR. The ideal case for all the scenarios would be that the flows share equal percentages of the available bandwidth. BBR is advertised as a fair TCP variant in terms of allocating the bandwidth to the other competing flows [23], [39]. However, this plot shows that BBR becomes too passive by trying to be fair against CUBIC even when it is being migrated. Moreover, it can be seen that during the period where congestion is absent, BBR cannot ramp up its sending rate to the link's capacity ($10\,\mathrm{Gbps}$). The peaks are constant around $5\,\mathrm{Gbps}$ link utilization. Also, after the CUBIC flow finishes, BBR continues to utilize only 50% of the available bandwidth. This indicates that the BDP calculation is not updated after the initialization phase, leading to inefficiency even after the competing flow vanishes. The inefficient bandwidth utilization of BBR and its fairness property causes its FCT to be much higher.

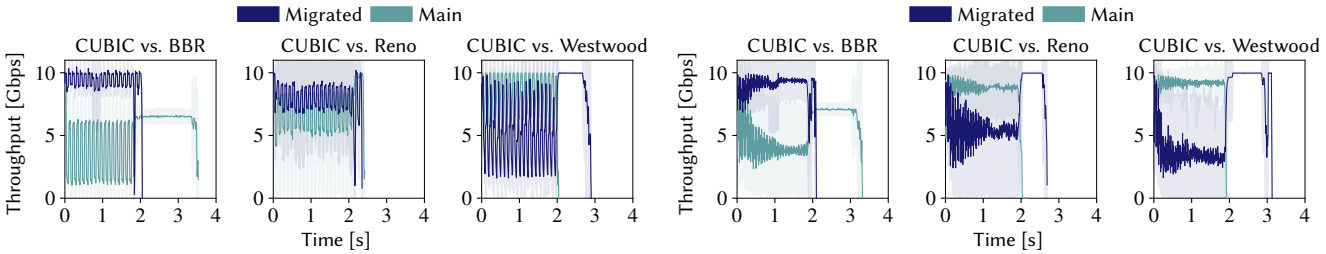
The competition between CUBIC and Reno, presented in the second subfigure of Fig. 6(a), has the fairest allocation among all scenarios. CUBIC and Reno, both loss-based CC schemes, behave similarly. Finally, Westwood (last sub-figure of Fig. 6(a)), a variant designed mainly for wireless, unreliable and lossy links, dominates the other variants in terms of throughput competition.

Since the CC algorithms of the variants are different from one another, their interaction exhibits different characteristics than the competition of the same variants. An aggressive congestion window growth rate leads to a higher share of the available bandwidth than the other variant. In contrast, a conservative congestion window growth rate may lead the flow to under-utilize the link.

Furthermore, when compared to a smaller reconfiguration period (presented in Fig. 6(b)), the flow behavior is observed to be similar. While the magnitude of the FCT difference is not the same, the overall behavior is similar. This indicates that in addition to the reconfiguration period, the interacting TCP variants also play a role in determining the effect on the FCT.

D. Reconfiguration Downtime

The scenarios so far considered an instantaneous reconfiguration of the circuits and re-routing of the flows. However, OCS generally incur a reconfiguration cost [17], [40], [41].



(a) Reconfiguration Period: 50ms.

(b) Reconfiguration Period: 10ms.

Fig. 6. Average throughput of one migrated and one static (main) flow. The first TCP variant in each title refers to the migrated flow and the second one indicates the main flow. Moreover, 95% confidence intervals are presented in the background with matching colors. In addition to the reconfiguration period, the mix of interacting TCP variants affects the achieved throughput as well.

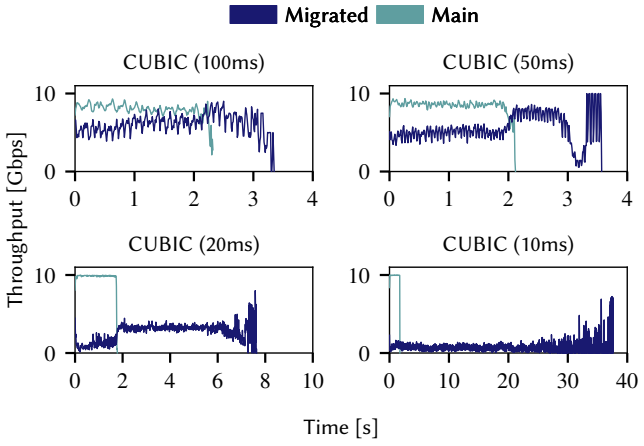


Fig. 7. Average throughput of TCP CUBIC flows with varying reconfiguration downtimes. Reconfiguration periods are reported in the title and the duty cycle is set to 10% in all scenarios. For instance, 10 ms reconfiguration downtime is created in case of a 100 ms reconfiguration period. Dropping packets during downtime increases the FCT of the migrated flow significantly.

This means that the circuit is destroyed during a reconfiguration process and the new circuit and link require some time to be set up. Packets arriving during the reconfiguration process are, in the worst case, dropped. Since the `Switch` does not drop any packets during reconfiguration, the reconfiguration downtime is emulated by inserting a rule to drop packets during the downtime.

For reconfiguration downtime analysis, two flows with 2 GB sizes using TCP CUBIC are considered. One of them is continuously migrated while the other one is served statically. The reconfiguration period is varied across scenarios and standardized 10% duty cycle is investigated, e.g. 50 ms *day time* and 5 ms *night time*.

Fig. 7 shows the average throughput of migrated and main CUBIC flows under different reconfiguration settings. Commercially available OCS have reconfiguration downtimes in the order of milliseconds. Accordingly, we evaluate values of 1, 2, 5 and 10 ms. The upper left sub-figure illustrates the scenario where the circuit is provisioned for 90 ms and the reconfiguration incurs a downtime of 10 ms. Unlike the zero downtime case, it can be observed that even at 100 ms reconfiguration period migrated flow achieves consistently lower throughput, and hence it translates into higher FCT for the migrated flow. The following sub-figures further display the low link utilization of the migrated flow.

The last sub-figure presents the most outstanding behavior where the main flow achieves over 95% of the bandwidth during congestion. After dropping packets, 9 ms does not allow enough time for TCP CUBIC to ramp up its sending rate. Therefore, the migrated flow is unable to create congestion that will reduce the throughput of the main flow. Moreover, it can also be seen that even after the congestion vanishes, the migrated flow suffers from migrations due to the reconfiguration downtime. In this case, reconfiguration downtime not only affects the migrated flow's FCT negatively, but it affects the main flow's FCT positively.

TABLE II
AVERAGE THROUGHPUT OF MIGRATED AND MAIN FLOWS.

Reconfiguration Period	Migrated Flow	Main Flow
50 ms	1.43 Gbps	1.24 Gbps
25 ms	1.39 Gbps	1.28 Gbps
10 ms	1.31 Gbps	1.27 Gbps

E. Small Flows and Random Arrivals

Empirical studies of DCNs show that over 80% of the flows are less than 10 KB and inter-arrival time (IAT) of flows are in the range of microseconds [42]. To improve scalability, RDCN designs such as [3], [4], [7] target rack-to-rack circuits. In such cases, the total rack-to-rack demand is considered and hence, groups of multiple flows of different sizes become subject to reconfigurations. In order to emulate this situation, we consider a scenario with flows in the range of 2^n MB for n in [0, 8]. The flows are separated into two groups: *migrated* and *main*. The flow volumes are randomly selected from the available volumes such that the total volume per group is 1 GB. The maximum allowed number of flows per group is set to 20. Flows arrive in pairs (one migrated and one main) with periods of 10 ms. All flows use TCP CUBIC.

Table II shows the average throughput of flows in each group under different reconfiguration periods. While the average throughput of the main flows varies less than 2%

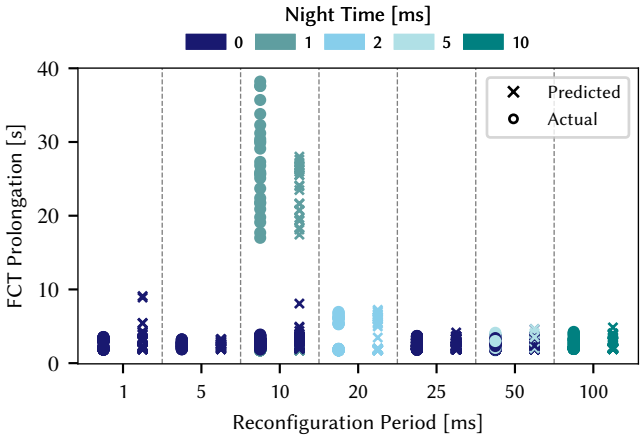


Fig. 8. Scatter plot of the predicted and actual FCT prolongations. The actual and predicted FCTs are plotted with different markers side-to-side with each other. The prediction is achieved with an MSE of $4.4\,\mathrm{s}^2$.

across scenarios, the migrated flows' throughput decreases as the reconfiguration period is decreased. Overall, this hints at the inefficient bandwidth utilization of flows when the reconfiguration period is decreased.

VI. Modeling

The analysis of the measurements in Sec. V showed that, in general, the migrated flow's FCT increases when the reconfiguration period decreases. This section extends the evaluations to build up an ML model to predict the effect of the reconfiguration period on the FCT prolongation of the migrated flow. The proposed model determines the factors of influence on FCT prolongation and formulates a prediction. The insights gained from the predictions of the model will serve as a tool to estimate the cost of reconfiguration scenarios on the FCT.

The FCT prolongation is defined as the time difference between the migrated flow's FCT and the main flow's FCT. The target variable to predict in this model is the FCT prolongation. The input variables for the model are the migrated TCP variant, the main TCP variant, *day time*, and the *night time*.

For better interpretability, a random forest model is used for training [43]. The training dataset consists of the FCTs measured from all of the scenarios introduced in Sec. IV.

The random forest model is optimized via a grid search of over 288 combinations of model parameters and evaluated using 10-fold cross-validation. The grid search returns the best parameters as 500 estimators, bootstrapping enabled, maximum tree depth of 100 with a minimum of two samples per leaf and two samples to split at each internal node.

Overall, the model predicts the FCT prolongation with a Mean Squared Error (MSE) of $4.4\,\mathrm{s}^2$. Moreover, the R^2 of the model is 0.88, which indicates that 88% of the variance in the results can be explained via this model. The unexplained variance is mainly related to the scenario with 10 ms reconfiguration period and 1 ms *night time*. Since the actual FCTs vary significantly, the model fails to extract its full potential.

Fig. 8 presents the scatter plot of the predictions of the best performing model and the actual FCT prolongations. The prolongation values range from no prolongation to as high as 40 s.

The prolongation effect is significant when the *night time* is 1 ms and the *day time* is 9 ms. Reconfiguration downtime has an important effect on FCT prolongation. However, the primary determinant is the *day time* in which a flow is served via the provisioned link. This effect is also shown clearly in the figure. At 2 ms *night time* with 20 ms *day time*, the FCT prolongation effect is much less.

The model predicts the effect of the reconfiguration period on the FCT prolongation with considerable performance. The availability of such a model enables tuning the RDCN architecture according to the TCP traffic mix.

VII. Conclusion

Achieving the full performance of RDCNs depend on TCP's ability to utilize the temporary high bandwidth links efficiently. Sub-millisecond reconfiguration periods pose a threat to the performance of state-of-the-art TCP implementations. The short duration of the high-bandwidth link provisioning periods hinders TCP's ability to ramp up its sending rate and leads to lower link utilization with higher FCTs.

This paper analyzed the most popular TCP variants' behavior under different reconfiguration scenarios. The findings indicate that reconfiguring the switch more frequently leads to a significant FCT increase on the migrated flow and the magnitude of this effect depends on the traffic mix. The proposed ML model generalizes the FCT prolongation and it will benefit the networking society to gain insights into the relationship between the reconfiguration scenario and FCT.

A generalized model that will serve as a foundation for predicting FCT prolongation with the incorporation of more input DCN parameters, e.g., switch buffer sizes and cable lengths is a direction for future work. The findings reported in this paper outline the benchmarks and pave the way for future work to design reconfiguration scenarios according to the analyzed traffic mix and the proposed ML model.

Acknowledgments

We acknowledge the financial support by the Federal Ministry of Education and Research of Germany (BMBF) in the program of "Souverän. Digital. Vernetzt." joint project 6G-life, project identification number 16KISK002 and Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 438892507.

References

[1] E. Jonas, J. Schleier-Smith, V. Sreekanti, C. Tsai, A. Khandelwal, Q. Pu, V. Shankar, J. Carreira, K. Krauth, N. J. Yadwadkar, J. E. Gonzalez, R. A. Popa, I. Stoica, and D. A. Patterson, "Cloud programming simplified: A berkeley view on serverless computing," *CoRR*, vol. abs/1902.03383, 2019. [Online]. Available: http://arxiv.org/abs/1902.03383

[2] V. Ziegler, H. Viswanathan, H. Flinck, M. Hoffmann, V. Räisänen, and K. Hätönen, "6g architecture to connect the worlds," *IEEE Access*, vol. 8, pp. 173 508–173 520, 2020.

[3] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat, "Helios: A hybrid electrical/optical switch architecture for modular data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, p. 339–350, Aug. 2010.

[4] C. Griner, J. Zerwas, A. Blenk, M. Ghobadi, S. Schmid, and C. Avin, "Cerberus: The power of choices in datacenter topology design-a throughput perspective," *POMACS*, vol. 5, no. 3, pp. 1–33, 2021.

[5] H. Liu, M. K. Mukerjee, C. Li, N. Feltman, G. Papen, S. Savage, S. Seshan, G. M. Voelker, D. G. Andersen, M. Kaminsky *et al.*, "Scheduling techniques for hybrid circuit/packet networks," in *Proc. 11th ACM CoNEXT*, 2015, pp. 1–13.

[6] M. Ghobadi, R. Mahajan, A. Phanishayee, N. Devanur, J. Kulkarni, G. Ranade, P.-A. Blanche, H. Rastegarfar, M. Glick, and D. Kilper, "Projector: Agile reconfigurable data center interconnect," in *Proc. ACM SIGCOMM 2016*. New York, NY, USA: Association for Computing Machinery, 2016, p. 216–229.

[7] W. M. Mellette, R. Das, Y. Guo, R. McGuinness, A. C. Snoeren, and G. Porter, "Expanding across time to deliver bandwidth efficiency and low latency," in *Proc. 17th USENIX NSDI*, 2020, pp. 1–18.

[8] K.-T. Foerster, M. Ghobadi, and S. Schmid, "Characterizing the algorithmic complexity of reconfigurable data center architectures," in *Proc. of the 2018 Symposium on Architectures for Networking and Communications Systems*, ser. ANCS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 89–96.

[9] M. K. Mukerjee, C. Canel, W. Wang, D. Kim, S. Seshan, and A. C. Snoeren, "Adapting TCP for reconfigurable datacenter networks," in *USENIX NSDI*. Santa Clara, CA: USENIX Association, Feb. 2020, pp. 651–666. [Online]. Available: https://www.usenix.org/conference/nsdi20/presentation/mukerjee

[10] M. T. Naing, T. T. Khaing, and A. H. Maw, "Evaluation of tcp and udp traffic over software-defined networking," in *2019 International Conference on Advanced Information Technologies (ICAIT)*, 2019, pp. 7–12.

[11] K. Miyazawa, S. Yamaguchi, and A. Kobayashi, "Performance evaluation of tcp bbr and cubic tcp in smart devices downloading on wi-fi," in *2020 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan)*, 2020, pp. 1–2.

[12] K. Ratna Pavani and N. Sreenath, "Performance evaluation of tcp-reno, tcp-newreno and tcp-westwood on burstification in an obs network," in *ADCOM*, 2012, pp. 19–24.

[13] K. Sasaki, M. Hanai, K. Miyazawa, A. Kobayashi, N. Oda, and S. Yamaguchi, "Tcp fairness among modern tcp congestion control algorithms including tcp bbr," in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, 2018, pp. 1–4.

[14] S. M. Irteza, A. Ahmed, S. Farrukh, B. N. Memon, and I. A. Qazi, "On the coexistence of transport protocols in data centers," in *Proc. IEEE ICC*, 2014, pp. 3203–3208.

[15] H. Liu, F. Lu, A. Forencich, R. Kapoor, M. Tewari, G. M. Voelker, G. Papen, A. C. Snoeren, and G. Porter, "Circuit switching under the radar with reactor," in *Proc. 11th USENIX NSDI*, 2014, pp. 1–15.

[16] H. Liu, M. K. Mukerjee, C. Li, N. Feltman, G. Papen, S. Savage, S. Seshan, G. M. Voelker, D. G. Andersen, M. Kaminsky *et al.*, "Scheduling techniques for hybrid circuit/packet networks," in *Proc. 11th ACM ConNEXT*, 2015, pp. 1–13.

[17] W. M. Mellette, R. McGuinness, A. Roy, A. Forencich, G. Papen, A. C. Snoeren, and G. Porter, "Rotornet: A scalable, low-complexity, optical datacenter network," in *Proc. ACM SIGCOMM*, 2017, p. 267–280.

[18] J. Postel, "Transmission control protocol," Sep 1981. [Online]. Available: https://rfc-editor.org/rfc/rfc793.txt

[19] M. Polese, F. Chiariotti, E. Bonetto, F. Rigotto, A. Zanella, and M. Zorzi, "A survey on recent advances in transport layer protocols," *CoRR*, vol. abs/1810.03884, 2018. [Online]. Available: http://arxiv.org/abs/1810.03884

[20] S. Ha, I. Rhee, and L. Xu, "Cubic: a new tcp-friendly high-speed tcp variant," *ACM SIGOPS operating systems review*, vol. 42, no. 5, pp. 64–74, 2008.

[21] V. Jacobson, "Congestion avoidance and control," *ACM SIGCOMM computer communication review*, vol. 18, no. 4, pp. 314–329, 1988.

[22] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "Tcp westwood: Bandwidth estimation for enhanced transport over wireless links," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, 2001, pp. 287–297.

[23] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: congestion-based congestion control," *Communications of the ACM*, vol. 60, no. 2, pp. 58–66, 2017.

[24] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center tcp (dctcp)," in *Proc. ACM SIGCOMM*, 2010, pp. 63–74.

[25] V. Addanki, O. Michel, and S. Schmid, "{PowerTCP}: Pushing the performance limits of datacenter networks," in *Proc. USENIX NSDI*, 2022, pp. 51–70.

[26] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," in *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, 2002, pp. 89–102.

[27] M. Alizadeh, A. Javanmard, and B. Prabhakar, "Analysis of dctcp: stability, convergence, and fairness," *ACM SIGMETRICS Performance Evaluation Review*, vol. 39, no. 1, pp. 73–84, 2011.

[28] T. Das and K. M. Sivalingam, "Tcp improvements for data center networks," in *2013 Fifth International Conference on Communication Systems and Networks (COMSNETS)*, 2013, pp. 1–10.

[29] A. Jain, A. Pruthi, R. Thakur, and M. Bhatia, "Tcp analysis over wireless mobile ad hoc networks," in *2002 IEEE International Conference on Personal Wireless Communications*. IEEE, 2002, pp. 95–99.

[30] K. G. Tsiknas, P. I. Aidinidis, and K. E. Zoiros, "Performance evaluation of transport protocols in cloud data center networks," *Photonic Network Communications*, 2021.

[31] J. Bennett, C. Partridge, and N. Shectman, "Packet reordering is not pathological network behavior," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 789–798, 1999.

[32] R. Cârpa, M. D. de AssunçÃo, O. Glück, L. LefÈvre, and J.-C. Mignot, "Evaluating the impact of sdn-induced frequent route changes on tcp flows," in *Proc. IEEE CNSM*, 2017, pp. 1–9.

[33] J. Zhang, K. Gao, Y. R. Yang, and J. Bi, "Prophet: Toward fast, error-tolerant model-based throughput prediction for reactive flows in dc networks," *IEEE/ACM Transactions on Networking*, vol. 28, no. 6, pp. 2475–2488, 2020.

[34] D. H. Hagos, P. E. Engelstad, A. Yazidi, and Ø. Kure, "General tcp state inference model from passive measurements using machine learning techniques," *IEEE Access*, vol. 6, pp. 28 372–28 387, 2018.

[35] V. Arun, M. T. Arashloo, A. Saeed, M. Alizadeh, and H. Balakrishnan, "Toward formally verifying congestion control behavior," in *Proc. ACM SIGCOMM*, ser. SIGCOMM '21, 2021, p. 1–16.

[36] V. GUEANT, "Iperf - the ultimate speed test tool for tcp, udp and sctptest the limits of your network + internet neutrality test." [Online]. Available: https://iperf.fr/

[37] "Netronome agilio cx isa-4000-10-2-2: 2x 10g sfp+ smartnic." [Online]. Available: https://stordirect.com/shop/adapter-cards/network-interface-cards/netronome-agilio-cx-isa-4000-10-2-2-2x-10g-sfp-smartnic/

[38] "Intel® ethernet network adapter x710 product specifications." [Online]. Available: https://ark.intel.com/content/www/us/en/ark/products/series/189530/intel-ethernet-network-adapter-x710.html

[39] B. Jaeger, D. Scholz, D. Raumer, F. Geyer, and G. Carle, "Reproducible measurements of tcp bbr congestion control," *Computer Communications*, vol. 144, pp. 31–43, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366419303470

[40] K.-T. Foerster and S. Schmid, "Survey of reconfigurable data center networks: Enablers, algorithms, complexity," *ACM SIGACT News*, vol. 50, no. 2, pp. 62–79, 2019.

[41] J. Zerwas, W. Kellerer, and A. Blenk, "What you need to know about optical circuit reconfigurations in datacenter networks," in *2021 33th International Teletraffic Congress (ITC-33)*, 2021, pp. 1–9.

[42] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *ACM SIGCOMM IMC*, 2010, pp. 267–280.

[43] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," *Electronics*, vol. 8, no. 8, 2019.