A Network Load Balancer for Geographically Distributed Datacenters

Lakshmi Krishnaswamy
CSE Department
University of California Santa Cruz
lakrishn@ucsc.edu

Katia Obraczka
CSE Department
University of California Santa Cruz
katia@soe.ucsc.edu

Abdul Kabbani
Microsoft
Mountain View, CA
akabbani@gmail.com

***Abstract*—Geographically distributed datacenters typically consist of datacenters that are located in different geographic regions and are connected through Wide-Area Networks (WANs). They service an array of applications that generate latency-sensitive traffic that typically is handled within a datacenter as well as traffic that has to be transmitted over WAN links. With the rapid increase in AI/ML training and inference tasks and cloud computing services, handling the coexistence and interaction between WAN- and DC traffic has become increasingly challenging. Some of the noteworthy challenges include the differences in DC- and WAN traffic quality of service (QoS) requirements, link utilizations and round-trip times. Existing network-level load balancing algorithms are not designed to handle the challenges that accompany the co-existence of DC-, and WAN traffic. We introduce Balancia, a transport agnostic, lightweight network load balancer that dynamically switches between per-flow and per-packet control in order to provide adequate performance for both intra-, and inter-DC traffic given their different characteristics and quality-of-service (QoS) requirements. We show that Balancia achieves close to 80% reduction in the 99% tail flow completion times on WAN traffic in the presence of DC-WAN coexistence, when compared against state-of-the-art load balancers. We also present different switch buffer designs in the context of DC-WAN architectures, to understand their impact on load balancing mechanisms.**

***Index Terms*—Data center networks, WAN, load balancing, low latency, congestion control**

I. Introduction

Cloud infrastructure providers have increasingly adopted geographic redundancy and physical diversity for their datacenters to better support the growing demand of cloud computing services for a wide-range of applications, while improving service availability, robustness and user perceived response time. Additionally, the need for cost-efficient access to energy sources has also been a driving force behind geo-distributed datacenters.

However, while interactions between datacenters are facilitated through high-bandwidth wide-area networks (WAN) [1], their co-existence with intra-datacenter workloads raise a range of challenges due to the contrasting characteristics and requirements between inter- (WAN) and intra-datacenter (DC) traffic. DC workloads are characterized by round-trip times (RTT) on the order of $\approx 100\mu$s, while WAN traffic exhibits RTTs on the order of up to 100ms. In addition, WAN networks are often constrained by their interconnect cost, as they are not as over-provisioned in terms of their bandwidth and resource capacity compared to DC networks, thus having relatively high bandwidth utilization. The ML boom exacerbated by the emergence of large language models has recently driven the need for workload distribution across datacenters at a scale never seen before, and is only projected to grow more in the coming years [2]. Consequently, as datacenter fabric and workload diversity increase, meeting application Quality of Service (QoS) requirements while efficiently utilizing datacenter resources (e.g., without over-provisioning) becomes even more challenging.

There has been extensive work on load balancers for purely DC workloads. Notably, state-of-the-art datacenter load balancers such as Vertigo [3] and Preemptive Deflection [4] have been shown to effectively reduce traffic tail latencies in the presence of microbursts, which are "microsecond scale congestion events" [3]. While microbursts are often present with datacenter networks, they are not characteristic of datacenter WAN traffic. Thus a scheme tailor-made for microbursts would not work optimally for WAN traffic as shown in the work [5]. Solutions such as in Annulus [1], Gemini [6] and Uno [2] have approached the problem from a purely end-to-end, congestion control perspective. They do not examine the challenges of path selection, or allocating DC- and WAN workloads across multiple paths to deliver the required QoS for each type of traffic. More importantly, with the coexistence of DC-WAN traffic, it is important to have both end-to-end, as well as in-network decision making capabilities. In the study [5], they highlight purely end-host based load balancing techniques, and purely in-network techniques. However, given the difference in time-scales for DC and WAN traffic, it is important to be able to sense, as well as react at both the end-host, as well as in-network level.

In this work, we introduce a novel load balancing mechanism that is transport agnostic, lightweight and dynamically switches between per-flow, and per-packet as well as end-host, and in-network decisions to provide adequate performance for DC-WAN mixed traffic while utilizing datacenter resources efficiently. The contributions of this paper can be summarized as follows:

- We describe our load balancer's design and implementa-

ISBN 978-3-903176-82-9

tion in detail, including its main components and mechanisms for congestion sensing and congestion-aware path-selection.
- We present a detailed quantitative and qualitative study exploring the effects of switch buffer design and Explicit Congestion Notification (ECN) markings for intra- and inter-datacenter workload interactions and understand their impacts on the performance of load balancing mechanisms.
- We develop experimental methodologies for modeling DC-WAN workload coexistence and plan to open-source our artifacts for the broader research community.
- We evaluate Balancia comparing its performance against state-of-the-art load balancers and highlight its consistent gains under DC-WAN mixed traffic scenarios. We show that it achieves close to 80% reduction in 99% tail flow completion times on WAN traffic in the presence of DC-WAN coexistence, when compared against state-of-the-art load balancers. For DC traffic, it achieves about similar performance to state-of-the-art load balancers, while still reducing close to 90% in the 99% tail flow completion times compared to the baseline ECMP [7] approach.

The rest of the paper is organized as follows: Section II summarizes load balancing background and Section III discusses related works that helps understand the current state-of-the-art. We introduce Balancia and describe its design in Section IV. Section V described our experimental methodology, including workload modeling and datacenter topology used in our experiments, and in Section VI, we present and discuss our experimental results. Finally, Section VII concludes the paper with a discussion of future research directions.

II. Background

Datacenter network load balancers help provide application flows with adequate QoS by assigning traffic to different network paths to prevent congestion build-up, as well as to improve overall network utilization. Consequently, network load balancing algorithms need to be congestion-aware, which means that they need to make use of different signaling mechanisms to perceive the onset of congestion, i.e., perform congestion sensing, as well as employ techniques to either proactively, or reactively respond to the congestion onset, i.e., congestion management. At their core, load balancer algorithms manage congestion by deciding how application flows share network resources. They use different signaling mechanisms to decide: (1) how to assign application to the available multiple network paths, (2) what granularity level the assignment is performed (e.g., flow, packet, packet aggregates) and (3) where that decision is made. In this section, these different design axes of network load balancers are discussed, providing the necessary background to network load balancing state-of-the-art and the current gaps that our work addresses.

A. Load Balancing Granularity

The granularity of the load balancing decision plays a major role in improving workload distribution and therefore in avoiding/mitigating congestion. There are three basic levels of granularity used by current load balancers, as follows:

Flow Level: In flow-based load balancing mechanisms [7]–[11], path assignment is done at a flow level, and thus each flow is assigned to a specific path. Upon sensing congestion, load balancers such as FlowBender [8] and PLB [9], reroute flows to less congested paths. The main advantage of such mechanisms is that it only requires end-hosts to participate in path selection and assignment without needing any in-network modifications. However, for low-entropy and high utilization workloads such as AI training workloads these mechanisms have been shown to be not as optimal [12].

Packet Level: Packet-level load balancers can distribute packets belonging to the same flow across multiple paths to effectively use redundant paths [3], [4], [13], [14]. However, this may cause packets to arrive out of order which translates into additional overhead at the transport layer. Alternatively, packet-level load balancers such as Vertigo [3] and Drill [13] include a shim layer that can arrange out-of-order packets before passing them to the transport layer. This results in additional overhead. In load balancers, where enqueued packets are extracted to provision load balancing, then there is also the need for hardware support as well.

Flowlet Level: Flowlet-level load balancing achieves a middle ground between per-flow and per-packet path defletion mechanisms [15]–[18]. Flowlets are defined as a group of packets belonging to the same flow and are detected when the packet gap between two successive packets exceeds a certain threshold, e.g., the maximal latency of all paths [15]. Thus, with flowlets, small bursts of packets can be assigned to a path, without incurring packet significant reordering overhead. However, the performance of flowlet-based techniques rely on precise setting of the inter-flowlet spacing [19].

B. Congestion Signals

Congestion-aware network load balancers can infer congestion from a number of signals which can be grouped into two main categories: “end-to-end” or “in-network” signals. The most common form of end-to-end signaling is based on estimated round trip times (RTTs). In-network signaling often relies on Explicit Congestion Notification (ECN) [20] markings as well switch queue occupancy. While packet loss acts as the primary source of congestion inference, incurring a packet loss can be very detrimental, particularly for latency sensitive workloads. Consequently, load balancers try to avoid packet loss events by proactively detecting the onset of congestion.

Explicit Congestion Notification (ECN) [20]: ECN markings enable switches to inform the sending host of congestion onset, even before a packet loss occurs. Whenever a switch experiences congestion, it modifies its IP header to inform the sending host. This is done by setting a bit in the traffic class field of the IP header [21]. The receiver sends back this marked ECN bit to the sender in its ACK packet header. The sender

then adjusts its sending rate in response to the congestion signal [21]. Works that make use of ECN markings for their congestion sensing mechanisms include [2], [8], [9], [11], [21].
Estimated Round Trip Time (RTT): Estimated RTT helps to understand end-to-end congestion. Based on how much the measured RTT deviates from the estimated RTT, the extent of congestion in the network can be inferred. This can be used as a signal either through the help of timing acknowledgments when they arrive at the sender sent by the receiver upon receiving data packets or through explicitly sending out small data packets and measuring the corresponding RTT. Works that make use of estimated RTTs for their congestion sensing include [11], [15], [16].
Switch queue length: Another approach to infer in-network congestion is through switch buffer occupancy and switch queue lengths. Onset of congestion is detected when switches receive more packets than their capacity to enqueue them. Load balancers such as [3], [4], [13]–[15] use this information to take action to prevent further congestion buildup.

III. Related Works

In the first part of this section, we review existing work on load balancing mechanisms for datacenter networks. The second part of the section describes some of the recent and ongoing efforts towards designing novel congestion control mechanisms for intra- and inter-datacenter communications.

A. Load Balancing in Datacenter Networks

There exists a plethora of work focusing on loading balancing for datacenter networks. However, they all focus on load balancing for traffic within the datacenter. Equal-Cost Multi-Path (ECMP) is one of the earlier approaches to network-level load balancing. It uses a hash function to determine a flow's outgoing path. ECMP's hash function uses a packet's header information, i.e., source and destination IP address and port numbers, to generate a hash key. The key is then used to find the corresponding path in the hash table. However, with ECMP, path collisions may result in uneven load distribution and frequent congestion events at relatively light utilization levels. Several load balancing works such as [3], [4], [8]–[11], [13]–[18], [22]–[24] have highlighted this major drawback with ECMP and devised more optimal load balancing designs. Thus, there have been different load balancers designed across the different granularities discussed in Section II, using the different congestion signals described. Some of the works that stand out and outperform most of the other load balancers, have been used for the quantitative analysis in Section VI.
FlowBender [8]: FlowBender is a flow-level load balancer that aims at guaranteeing congestion-aware decisions. It has been designed as an end-host-based solution that does not involve any switch modifications. FlowBender uses explicit congestion notification (ECN) to detect congestion. FlowBender's successor "PLB" [9] has been successfully deployed at Google, and has been shown to significantly improve overall network utilization and reduce tail latencies.
DIBS [14]: DIBS or Detour Induced Buffer Sharing uses per-packet deflection as its load balancing approach. Whenever a switch receives more packets than it can enqueue, it detours the excess packets to a randomly selected port with enough buffer space. DIBS' hot potato type technique performs well under light loads, but struggles on increased network load conditions.
DRILL [13]: Distributed Randomized In-Network Localized Load Balancing (DRILL) makes per-packet level decisions using local information available at each switch to distribute load as evenly as possible on a microsecond timescale. It makes load-aware forwarding decisions local to each switch. However, this is prone to more packet reorderings, and requires stateful switch processing [9]. It has also been observed to be not suitable for asymmetric network conditions and where switch queue occupancies are uneven, as described in Section VI.
Vertigo [3]: Vertigo employs an in-network, host-assisted, selective deflection solution. In particular, packets that potentially cause persistent congestion in the network are selected for deflection, thereby reducing the reordering that deflection in general can cause. However, this requires extracting packets already enqueued in the switch buffer, which isn't supported in current datacenter switches [4].
Preemptive Deflection [4]: To address Vertigo's limitation, Preemptive Deflection was introduced as its successor. Preemptive deflection uses an admission control technique that evaluates if an incoming packet should be deflected or enqueued. This decision is made based on priorities assigned to the packets based on the current packet, enqueued packets, and the destination's queue occupancy. Based on this, packets that are more prone to cause long-lasting congestion are deflected. We plan to include evaluations of this as part of our future work.
REPS [21]: Recycled Entropy Packet Spraying for Adaptive Load Balancing and Failure Mitigation (REPS) is a very recent work, that is a per-packet load balancing algorithm that is designed to optimize network utilization along with rapid recovery from link failures. It achieves this through the caching of good, uncongested paths, in a circular buffer. However, it does require out-of-order support from the transport layer, thus not transport-agnostic.

B. Congestion Control for Intra- and Inter-Datacenter Communication

Works such as Uno [2], Annulus [1], and Gemini [6] highlight the growing trend of inter-datacenter networks that results in DC-WAN traffic coexistence. In Annulus [1], a dual control loop makes congestion control decisions to explicitly address the diverse QoS needs of mixed DC-WAN traffic. Gemini [6] uses a combination of Explicit Congestion Notification (ECN) and RTT signals to make congestion control decisions. These works address how congestion control mechanisms can adapt to DC-WAN scenarios. Uno [2] highlights the need for co-designing congestion control and load balancing algorithms for intra- and inter-DC communications. However, all of these mechanisms such as Annulus [1], Gemini [6] and Uno [2],

make purely end-host decisions for congestion control. However, with the coexistence of DC-WAN traffic, it is important to have both end-host, as well as in-network decision making capabilities in order to optimally operate at the different timescales.

C. Discussion

In this section, we covered existing work on load balancing mechanisms for datacenter networks, as well as the ongoing efforts towards designing novel congestion control mechanisms for the coexistence of DC-WAN traffic. While there has been extensive work for network load balancers for datacenter networks, they all perform either purely end-host or purely in-network decision for load balancing, which has also been observed in the work [5]. Additionally, the discussed congestion control efforts also perform purely end-host based decisions, even for DC-WAN mixed traffic conditions. However, given the different timescales both DC, and WAN operate in and their varied latency and throughput requirements, it is important to be able to have separate control loops, and being able to dynamically switch between end-host and in-network decision making to work optimally for both DC and WAN traffic.

IV. Balancia

In Section III-C, we highlight the importance of a dual control and decision making mechanism for network load balancers in order to best provision for DC-WAN mixed traffic, given their different requirements and operation timescales.

Balancia, our novel network load balancer, is able to adapt to both intra- and inter-DC traffic and address their QoS requirements by sensing congestion and making decisions accordingly at the appropriate time scales. It dynamically switches between different granularity of control in order to provide adequate performance for both intra- and inter-DC traffic. At its core, Balancia is a distributed, load balancing technique that can dynamically switch between end-host and in-network decision making. Additionally, it provides in-network support for congestion sensing and management. Similar to other load balancing techniques, Balancia has three main components:

- Path Selection
- Congestion Sensing
- Congestion Management

Figure. 1 illustrates Balancia's decision flow - at the end-host, Balancia decides on if the particular flow is of DC or WAN type. When there is a DC flow at the end host, Balancia uses Vertigo's packet-level forwarding technique, inspired by the power of two choices [25] mechanism. This makes in-network decisions on packet-by-packet forwarding. When there is a WAN flow at the end host, Balancia maps the flow to a path determined by the hash computation of five fields, namely the source port, destination port, source IP, destination IP and the Time-To-Live (TTL) field, as done in FlowBender. In particular, this makes end-host decisions on flow-level assignment. In this manner, depending on the

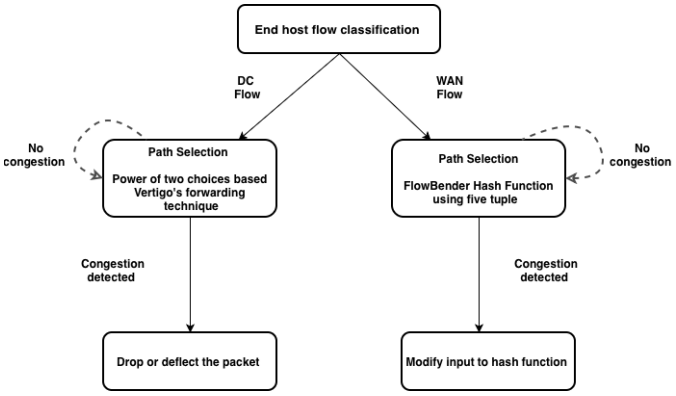

Fig. 1. Balancia: Components and Workflow

type of flow, i.e., DC or WAN, Balancia dynamically switches between end-host for WAN flows and in-network for DC flows decision making for path selection, at either packet- (for DC flows) or flow level (for WAN flows). Additionally, it makes use of both end-to-end, as well as in-network congestion signals. With Vertigo, it infers congestion through in-network switch occupancy signaling, along with host assistance in order to understand which flow can lead to more persistent congestion in the network. With FlowBender, there is a set congestion threshold "T", which is determined empirically as described in FlowBender's original paper [8]. Whenever the ECN markings exceed this threshold, the input to the hash function is modified, thus re-assigning the flow to a new path, different from the current, congested path. These are particularly important decision and control mechanisms in order to react in-time for the DC flows, as well as wait for the end-to-end signaling to react accurately for WAN flows.

In summary, Balancia switches between Vertigo and FlowBender for DC and WAN flows respectively. In particular, by devising a "dual" approach to load balancing via switching between flow level and packet level, Balancia prevents congestion buildup in the network core thus resulting in much lower tail latencies even for large flows, as shown by our experimental results presented in Section VI. This design has also been motivated by the performance of both Vertigo and FlowBender when evaluated with DC and WAN workloads. In particular, it was observed that Vertigo performs the best with DC traffic, while FlowBender's per-flow based path allocation and flow level rerouting works well with WAN flows that traversed long haul links.

V. Experimental Methodology

This section presents the experimental setup and methodology we used to carry out the comparative performance evaluations of Balancia against state-of-the-art load balancers which are described in Section III. In the remaining of this section, we describe the simulation platform we used in our experiments, the DC-WAN topology model as illustrated in Figure. 2 and recommended in [5], as well as the DC-WAN

workload models representative of the different geo-distributed datacenter workloads.

A. Experimental Platform

All simulations were carried out using the OMNeT++ simulator version 5.6.2 [26] and INET framework version 4.3.2 [27]. The CloudLab [28] platform was used to carry out the experiments. We use the open-source Vertigo codebase [3] for validating implementation of load balancers Vertigo [3], DIBS [14] and DRILL [13].

B. Topology Modeling

For evaluating the performance of Balancia and benchmarking against state-of-the-art load balancers, we use the WAN topology as shown in Figure. 2. This was recommended in [5], and closely models after popular works such as [1] and [2]. In the DC-WAN topology model (see Figure. 2), the network consists of two datacenter fabrics interconnected through their core switches. Each datacenter fabric has 4 pods, each of which has 4 edge switches, 4 aggregate switches and 16 servers, thereby the whole network consists of 128 servers in total. Core switches in one datacenter fabric are connected via WAN links to the core switches of the other fabric. In order to have the WAN links oversubscribed, each core switch in a given datacenter fabric connects to every core switch in the other datacenter fabric, thus forming a fully connected mesh. We set all links within a datacenter fabric to a uniform speed of 100Gbps. The links between the core switches are set to 10Gbps. The links between the top-of-rack (ToR) switches and the server have a delay of 1μs , while the links between the aggregate switches and ToRs have a delay of 2μs. The WAN links, between the core switches are set to have a delay of 1ms [1], [5]. The bottleneck links for DC traffic are the links between the aggregate and the core switches. The bottleneck links for WAN traffic are the inter-DC links, or WAN links. The flow inter-arrival times are scaled to vary the supplied load in all the experiments.

C. Workload Modeling

WAN Workloads: In order to capture the high utilization of WAN links in a WAN network [1], we use long flows of 10MB to mimic WAN Traffic. We generate around 12,800 flows.

DC-WAN Mixed Workloads: To model the coexistence of DC and WAN traffic in geo-distributed datacenters, we use both synthetic as well as realistic web search workloads to represent DC traffic within DC Fabric 1 and DC Fabric 2. For the synthetic workloads, we follow the recommendations from [5]. It consists of a bimodal distribution, that overcomes the issue of long run times while still capturing the heavy-tail distribution of typical datacenter network workloads. It consists of short flows that are $50KB$ in size and large flows that are $1MB$ in size. These were chosen, in order to maintain reasonable simulation times, while capturing the heavy-tailed distribution of DC workloads. In order to mimic the heavy-tailed distribution of DC workloads, 90% of the total flows are $50KB$, while the remaining 10% are of $1MB$. We generate around $51,200$ flows. This value was chosen for the number of flows, given that it was sufficient to be representative of the heavy-tailed distribution. For the realisitic workloads, we sample flow sizes from the cumulative distribution function of web search workloads [29], to represent DC traffic within DC Fabric 1 and DC Fabric 2. For the WAN traffic, which is transmitted over the WAN links between the two DC fabrics, we model a 1:5 flow ratio of WAN to DC traffic, which is based on recommendations from [2], [1] and [5]. The results with the synthetic workloads have been omitted due to space constraints, but they followed the same trend with results of the realistic workloads, which have been elaborated in Section. VI.

D. Experimental Setup

Transport Protocol Parameters: DCTCP has been the recommended transport protocol for datacenter networks. It is also used in most related works such as [15], [14], [13], [3], [4]. Thus, we use DCTCP [30] as the transport protocol for all our experiments. As recommended in the DCTCP paper [30], the per-port switch buffer capacity is set to be equal to the bandwidth delay product of the specific datacenter network. However it is important to note here that given that Balancia is transport agnostic, other transport protocols can also be used alongside, and Balancia does not depend on DCTCP for it's performance. Both the low ECN marking $K_{\min}$ and high ECN marking $K_{\max}$ thresholds are set to be equal [30]; to one-third of the per-port switch buffer capacity . The values used for these parameters in our experiments are described in Section VI and summarized in Table. II and Table. III . Additionally, to avoid the effects of TCP's slow-ramp up, we set the initial congestion window size to be 10 packets. Given the high bandwidth-delay product associated with the high link speeds, in order to maximize throughput, window scaling is also enabled, thus preventing the advertised window from creating a bottleneck. These values can be found in Table. I.

Load Balancer Specific Parameters: Because Vertigo and DRILL employ a reordering layer to mitigate the effects of packet reordering caused by their route deflection technique, they use a parameter known as the *ReOrdering Timer*. This timer is defined as the "maximum duration of time that the ordering component waits for a single delayed packet before starting to process out-of-order packets" [3]. Vertigo recommends setting this value to the time it takes for a single packet to traverse a network with almost full buffers. For our WAN, and DC-WAN experiments, we use the recommendations provided in [5]. FlowBender has a congestion threshold parameter "T" which is set in order to alleviate congestion at its early onset. Whenever the ECN markings exceed this threshold, the input to the hash function is modified, thus assigning the flow to an uncongested path. We follow the recommendations provided in [8] to set this value.

Performance Metrics: In our results, we report mean and

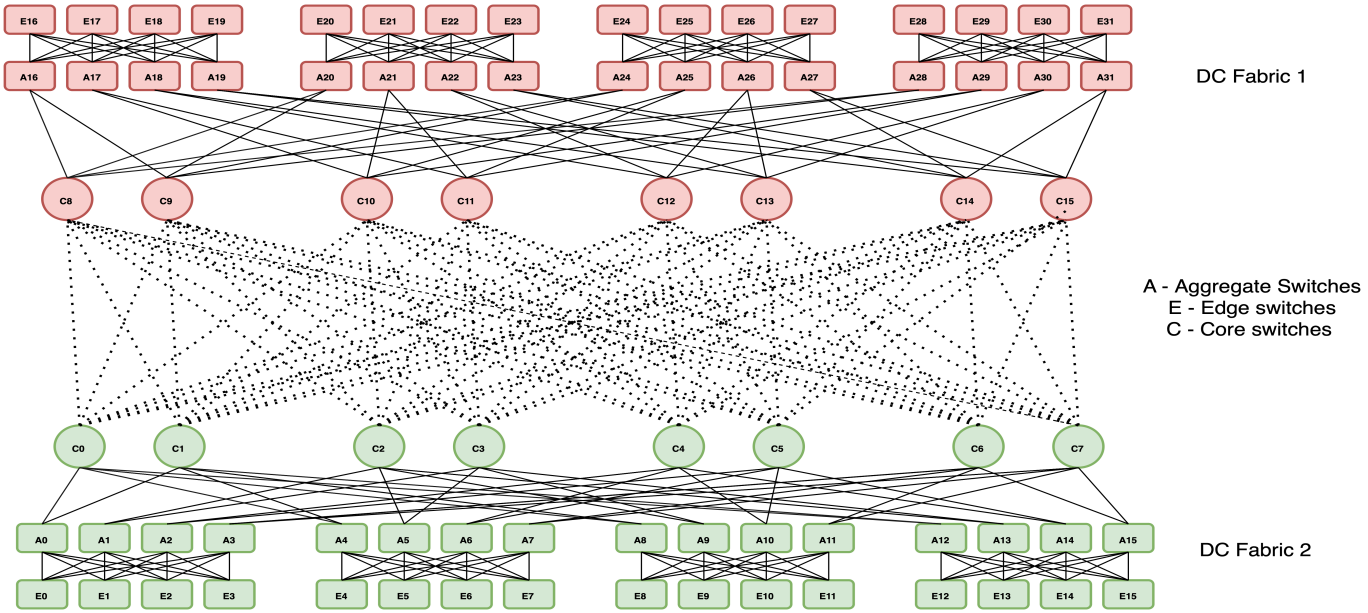

Fig. 2. WAN Network Topology

99% flow completion times (FCTs). Looking at 99% FCT helps establish how long the last 1% of the workload takes to complete, while the mean FCT helps understand the overall effectiveness of a given load balancer.

VI. Evaluation and Results

This section describes the experiments carried out evaluating Balancia under different scenarios.We evaluate Balancia and state-of-the-art load balancers under purely WAN workloads to explore the impact of switch buffer capacity design. We then evaluate the performance of Balancia and state-of-the-art load balancers under the co-existence of realistic DC and WAN workloads.

A. Exploring Impact of Switch Buffer Capacities

DC workloads have much shorter RTTs (20μs), and are latency sensitive, when compared with the WAN workloads. While the WAN workloads, have longer RTTs ($2ms$) and high link utilization. This creates an interesting design challenge for switch buffer sizing. We experimented with the following :

- Uniform switch buffer capacities of 1MB, at the edge, aggregate and core switches. These values can aslo be found in Table. III. This was based on the recommendations in [5].
- Non-uniform switch buffer capacities, also described in this work as "heterogeneous switch buffer capacities", where the edge and aggregate switches are set to be shallow, and the core switches are set to be deeper, with the exact values described in Table. II.

The latter design choice has been motivated given that DC workloads have tight latency requirements, and thus having deep switch buffer - that are significantly larger than the bandwidth delay product can cause buffer bloat and queue buildups that can further lead to transient congestion pockets. Thus, the aggregate and edge switch capacities have been fixed to be equal to the bandwidth delay product of the datacenter network. However, core switches interconnect the two fabrics, and their interconnects carry the WAN traffic. Given the high throughput requirements of WAN, having shallow buffers similar to the edge and aggregate switches will lead to quick overflow of buffer due to the bandwidth mixmatch. Additionally, with the increase of AI/ML training as an inter-communication, works like [31], highlight their datacenter networks being designed with deep core switches.

The WAN workloads described in the Section. V-C are used in the network Topology (see Figure. 2), and the experimental values are setup as described in the Section V-D, and listed in Table. I. The results plotted in the graphs show 99% FCT as a function of the load across the bottleneck links. Looking at 99% FCT helps establish how long the last 1% of the workload takes to complete.

TABLE I
Experimental Parameters

Parameter	Value
Initial Congestion Window Size	10 packets
Initial RTO	$1s$
Minimum RTO	$100ms$
Vertigo and DRILL ReOrdering Timer	$0.002038s$
FlowBender's congestion threshold	5%

Results and Discussion: From Figure. 3 we see that ECMP, being congestion agnostic, causes collisions due to multiple flows being assigned the same path under high load. DIBS' approach of deflecting packets to any switch with available buffer, when the next hop buffer is full; causes packet reordering and significant performance degradation under high loads.

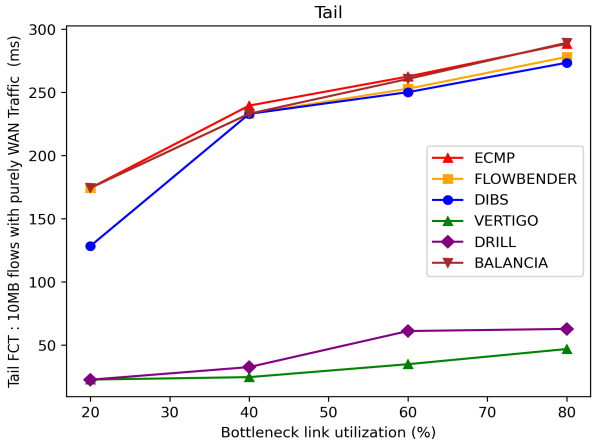

Fig. 3. WAN Workloads - 99% FCT for 10MB flows with homogeneous switch capacities

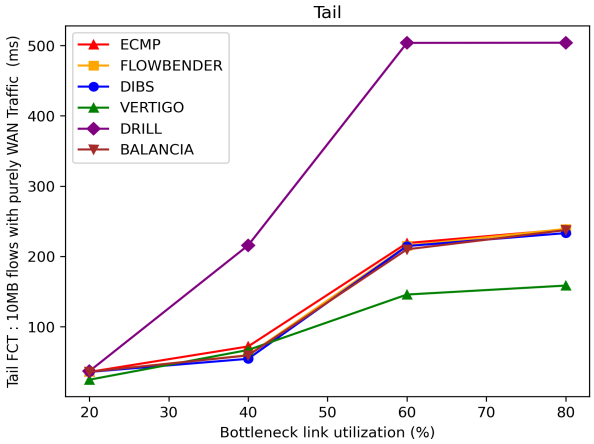

Fig. 4. WAN Workloads - 99% FCT for 10MB flows with heterogeneous switch capacities

TABLE II
HETEROGENEOUS SWITCH BUFFER PARAMETERS

Parameter	Value
Edge Switch Per-Port Capacity	$250KB$
ECN Marking Threshold for Edge Switches	56 packets
Aggregate Switch Per-Port Capacity	$250KB$
ECN Marking Threshold for Aggregate Switches	56 packets
Core Switch Per-Port Capacity	$7MB$
ECN Marking Threshold for Core Switches	1555 packets

TABLE III
UNIFORM SWITCH BUFFER PARAMETERS

Parameter	Value
All switches Per-Port Capacity	$1MB$
ECN Marking Threshold at all switches	223 packets

However, Vertigo and DRILL still outperform both Balancia and FlowBender. However, it is important to note that this is with buffer capacities of 1MB at all switches, and so this does not scale well with bigger topologies. From Figure. 4, when using switch design as outlined in Table. II, we notice that DRILL's 99% FCT is significantly worse compared to rest of the load balancers. This is due to its purely local, per-hop decisions. Which can attribute entire path's congestion inference to just the local switch occupancy, which in the case of heterogeneous switch buffer designs, or any asymmetric network doesn't always hold true.

B. DC-WAN Mixed Workloads

In these DC-WAN mixed workload experiments, we study the performance of Balancia alongside state-of-the-art load balancers, using the network topology illustrated in Figure 2. In particular, we examine performance with the web search workloads described in Section. V-C. The parameters used in these experiments are the heterogeneous switch capacity settings described Table. II. Rest of the parameters are set as described in Table. I. In order to better understand the results, we classify the results into different flowsize groups. The different flow groups can be defined as :

- $<$ 10KB DC flows
- $[10\text{KB}, 300\text{KB})$ DC flows
- $[300\text{KB}, 10\text{MB})$ DC flows
- 10MB WAN flows

The x-axis represents the load on the bottleneck links, i.e., the supplied DC load. The y-axis represents mean FCT, and 99% FCT. We vary the utilization over the WAN links, to examine the effects on the DC and WAN traffic due to their co-existence. We have included the results with 80% WAN link utilization graphs for discussion. This is also illustrative of WAN traffic's high utilization, as described in [5], [1], [6] and [2]. Each experiment was run three times by varying the random seed for generating different inter-arrival times and destination servers for the workloads. In particular, we have added the standard deviation bars across the multiple runs for these experiments.

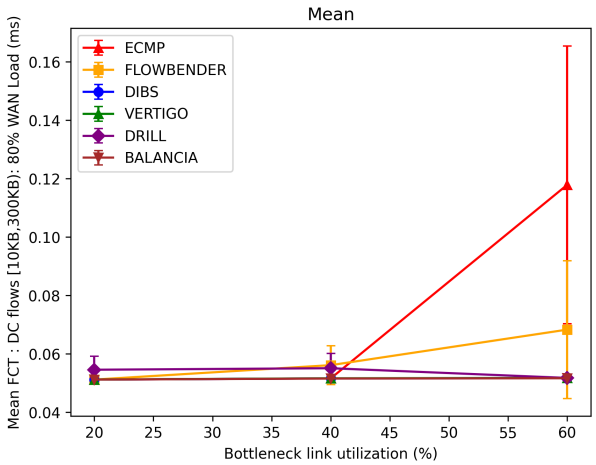

Fig. 5. Mean FCT for [10KB,300KB) flows with 80% WAN Load

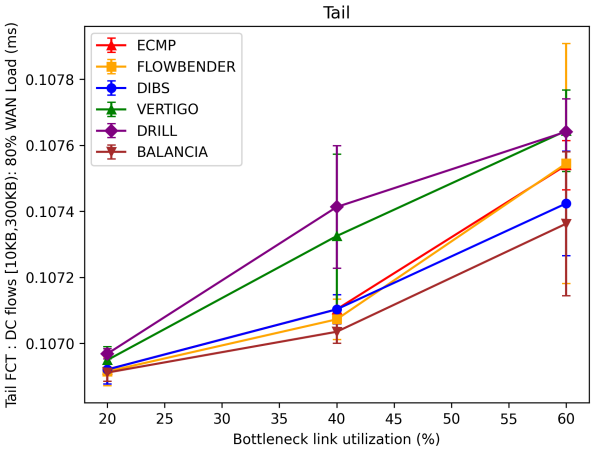

Fig. 6. 99% FCT for [10KB,300KB) flows with 80% WAN Load

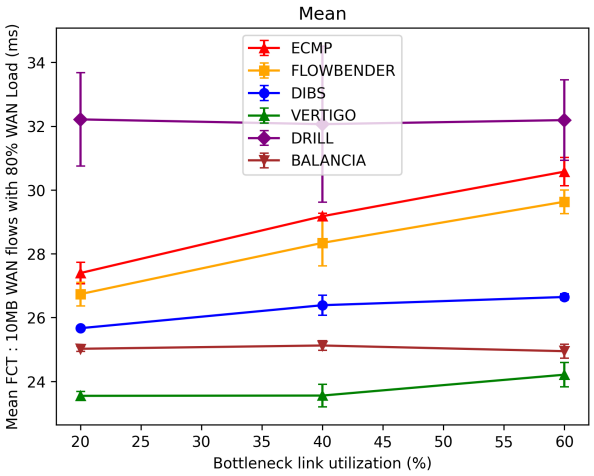

Fig. 7. Mean FCT for 10MB flows with 80% WAN Load

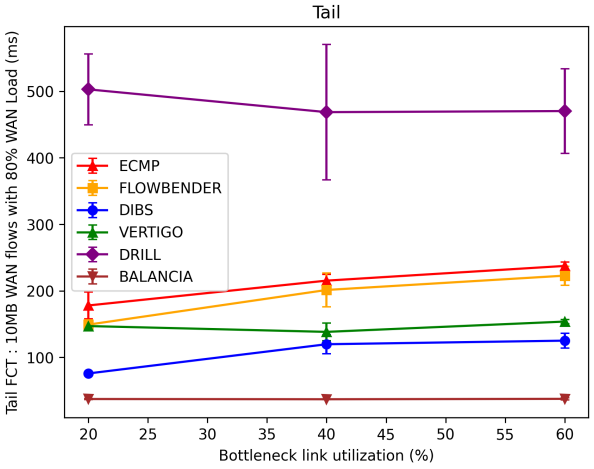

Fig. 8. 99% FCT for 10MB flows with 80% WAN Load

TABLE IV
RELATIVE 99% FCT REDUCTION (%) OVER ECMP AT 80% DC LOAD AND 80% WAN UTILIZATION

Load Balancer	<10 KB	[300 KB, 10 MB)	10 MB
Balancia	**0.78**	**99.10**	**85.05**
Vertigo	**0.78**	99.09	38.00
FlowBender	0.21	98.31	4.33
DRILL	0.65	99.04	-105.90
DIBS	0.01	99.00	36.55

Results and Discussion: From Figure. 5 and Figure. 6, we observe that Balancia exhibits consistently low FCTs for DC flows under 80% high WAN utilization. DRILL being a local, per-hop based load balancing mechanism, it uses merely the local switch sizes to make the next hop decisions, thus being oblivious to the rest of the path conditions. This kind of decision making is quick to cause high tail latencies when we have non-uniform switch buffer capacities. From Figure. 7 and Figure. 8 for the WAN flows, while the mean FCT of Vertigo and Balancia is comparable, the 99% FCT shows how Balancia has a significantly lower 99% FCT than Vertigo, with about 78.6% reduction in 99% FCT. With Balancia's dual approach to load balancing, achieved by switching at both flow and packet level, as well as through end-host and in-network decisiong making, we are able to better manage the congestion buildup. By combining congestion signals at different granularities, Balancia is able to lower FCTs even under high load, for both DC and WAN traffic. Vertigo by itself does not perform optimally in particular due to its packet deflections mechanism, which does not add value in networks with heterogeneous switch buffer capacities.

We present in Table. IV, the relative reduction of 99% FCT (%) over ECMP, with 80% supplied DC load and 80% utilization of WAN links. We use ECMP as the baseline to show the relative performances, as it is one of the more earlier, and simpler approaches. From the table, we see Balancia's consistent low mean, and 99% FCT for all flowsizes. DRILL's 99% FCT depicts a negative value (Given that it's FCT is higher than ECMP), due it its instability at high loads. This as well is attributed to its local decision, which specifically with the co-existence of DC-WAN flows on heterogeneous switch capacities is quick to fall into instability.

VII. CONCLUSION AND FUTURE WORK

With the proliferation of AI workloads, and distributed training, there is a growing need to service the co-existence of DC and WAN traffic within a datacenter. Based on previous work [5], there is an existing gap in the load balancing space that addresses this co-existence of DC-WAN traffic. In this work, we present Balancia, a lightweight, transport-agnostic network load balancer, that dynamically switches between per-flow and per-packet control decisions to provide optimal performance for DC-WAN mixed traffic. Further, Balancia has both end-host, and in-network decision making capabilities, to support the different timescales of DC and WAN traffic.

Through detailed quantitative evaluations, we exhibit Balancia's optimal performance, under different workload scenarios. In particular, we show that Balancia achieves close to 80% reduction in 99% tail flow completion times. As part of our future work, we plan to model AI/ML training workloads across datacenters, and evaluate Balancia's performance for optimizing AI/ML training at scale, across datacenters.

REFERENCES

[1] A. Saeed, V. Gupta, P. Goyal, M. Sharif, R. Pan, M. Ammar, E. Zegura, K. Jang, M. Alizadeh, A. Kabbani, and A. Vahdat, "Annulus: A dual congestion control loop for datacenter and wan traffic aggregates," in *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 735–749. [Online]. Available: https://doi.org/10.1145/3387514.3405899

[2] T. Bonato, S. Abdous, A. Kabbani, A. Ghalayini, N. Gebara, T. Lam, A. Agarwal, T. Chen, Z. Yu, K. Taranov, M. Elhaddad, D. De Sensi, S. Ghorbani, and T. Hoefler, "Uno: A One-Stop Solution for Inter- and Intra-Data Center Congestion Control and Reliable Connectivity," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. St. Louis MO USA: ACM, Nov. 2025, pp. 1195–1210.

[3] S. Abdous, E. Sharafzadeh, and S. Ghorbani, "Burst-tolerant datacenter networks with Vertigo," in *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*. Virtual Event Germany: ACM, Dec. 2021, pp. 1–15.

[4] ——, "Practical Packet Deflection in Datacenters," *Proceedings of the ACM on Networking*, vol. 1, no. CoNEXT3, pp. 1–25, Nov. 2023.

[5] L. Krishnaswamy, K. Obraczka, and A. Kabbani, "What's Next for Datacenter Load Balancers - A Comparative Performance Study of the State-of-the-Art," in *2024 IEEE 13th International Conference on Cloud Networking (CloudNet)*, Nov. 2024, pp. 1–9.

[6] G.Zeng, W.Bai, G.Chen, K.Chen, D.Han, Y. Y.Zhu, and L.Cui, "Congestion control for cross-datacenter networks," *IEEE/ACM Trans. Networking*, vol. 30, no. 5, pp. 2074–2089, Oct. 2022.

[7] C. Hopps *et al.*, "Analysis of an equal-cost multi-path algorithm," RFC 2992, November, Tech. Rep., 2000.

[8] A. Kabbani, B. Vamanan, J. Hasan, and F. Duchene, "FlowBender: Flow-level Adaptive Routing for Improved Latency and Throughput in Datacenter Networks," in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. Sydney Australia: ACM, Dec. 2014, pp. 149–160. [Online]. Available: https://dl.acm.org/doi/10.1145/2674005.2674985

[9] M. A. Qureshi, Y. Cheng, Q. Yin, Q. Fu, G. Kumar, M. Moshref, J. Yan, V. Jacobson, D. Wetherall, and A. Kabbani, "PLB: congestion signals are simple and effective for network load balancing," in *Proceedings of the ACM SIGCOMM 2022 Conference*, ser. SIGCOMM '22. New York, NY, USA: Association for Computing Machinery, Aug. 2022, pp. 207–218.

[10] K. He, E. Rozner, K. Agarwal, W. Felter, J. Carter, and A. Akella, "Presto: Edge-based load balancing for fast datacenter networks," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, ser. SIGCOMM '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 465–478. [Online]. Available: https://doi.org/10.1145/2785956.2787507

[11] H. Zhang, J. Zhang, W. Bai, K. Chen, and M. Chowdhury, "Resilient datacenter load balancing in the wild," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 2017, pp. 253–266.

[12] K. Qian, Y. Xi, J. Cao, J. Gao, Y. Xu, Y. Guan, B. Fu, X. Shi, F. Zhu, R. Miao, C. Wang, P. Wang, P. Zhang, X. Zeng, E. Ruan, Z. Yao, E. Zhai, and D. Cai, "Alibaba HPN: A Data Center Network for Large Language Model Training," in *Proceedings of the ACM SIGCOMM 2024 Conference*. Sydney NSW Australia: ACM, Aug. 2024, pp. 691–706.

[13] S. Ghorbani, Z. Yang, P. B. Godfrey, Y. Ganjali, and A. Firoozshahian, "DRILL: Micro Load Balancing for Low-latency Data Center Networks," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. Los Angeles CA USA: ACM, Aug. 2017, pp. 225–238.

[14] K. Zarifis, R. Miao, M. Calder, E. Katz-Bassett, M. Yu, and J. Padhye, "DIBS: just-in-time congestion mitigation for data centers," in *Proceedings of the Ninth European Conference on Computer Systems - EuroSys '14*. Amsterdam, The Netherlands: ACM Press, 2014, pp. 1–14. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2592798.2592806

[15] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V. T. Lam, F. Matus, R. Pan, N. Yadav *et al.*, "Conga: Distributed congestion-aware load balancing for datacenters," in *Proceedings of the 2014 ACM conference on SIGCOMM*, 2014, pp. 503–514.

[16] N. Katta, M. Hira, C. Kim, A. Sivaraman, and J. Rexford, "Hula: Scalable load balancing using programmable data planes," ser. SOSR '16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2890955.2890968

[17] N. Katta, M. Hira, A. Ghag, C. Kim, I. Keslassy, and J. Rexford, "Clove: How i learned to stop worrying about the core and love the edge," in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, ser. HotNets '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 155–161. [Online]. Available: https://doi.org/10.1145/3005745.3005751

[18] E. Vanini, R. Pan, M. Alizadeh, P. Taheri, and T. Edsall, "Let it Flow: Resilient Asymmetric Load Balancing with Flowlet Switching."

[19] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 2, pp. 51–62, Mar. 2007.

[20] S. Floyd, K. K. Ramakrishnan, and D. L. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," Internet Engineering Task Force, Request for Comments RFC 3168, Sep. 2001.

[21] T. Bonato, A. Kabbani, A. Ghalayini, M. Papamichael, M. Dohadwala, L. Gianinazzi, M. Khalilov, E. Achermann, D. D. Sensi, and T. Hoefler, "REPS: Recycled Entropy Packet Spraying for Adaptive Load Balancing and Failure Mitigation," Sep. 2025.

[22] D. Wetherall, A. Kabbani, V. Jacobson, J. Winget, Y. Cheng, C. B. Morrey Iii, U. Moravapalle, P. Gill, S. Knight, and A. Vahdat, "Improving Network Availability with Protective ReRoute," in *Proceedings of the ACM SIGCOMM 2023 Conference*. New York NY USA: ACM, Sep. 2023, pp. 684–695.

[23] H. Xu and B. Li, "TinyFlow: Breaking elephants down into mice in data center networks," in *2014 IEEE 20th International Workshop on Local & Metropolitan Area Networks (LANMAN)*. Reno, NV: IEEE, May 2014, pp. 1–6.

[24] Y. Geng, V. Jeyakumar, A. Kabbani, and M. Alizadeh, "Juggler: a practical reordering resilient network stack for datacenters," in *Proceedings of the Eleventh European Conference on Computer Systems*, ser. EuroSys '16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2901318.2901334

[25] M. Mitzenmacher, A. W. Richa, and R. Sitaraman, "The Power of Two Random Choices: A Survey of Techniques and Results," in *Handbook of Randomized Computing*, D.-Z. Du, P. M. Pardalos, S. Rajasekaran, P. M. Pardalos, J. H. Reif, and J. Rolim, Eds. Boston, MA: Springer US, 2001, vol. 9, pp. 255–312.

[26] "Omnet++ discrete event simulator," https://omnetpp.org.

[27] "Inet framework," https://inet.omnetpp.org.

[28] D. Duplyakin, R. Ricci, A. Maricq, G. Wong, J. Duerig, E. Eide, L. Stoller, M. Hibler, D. Johnson, K. Webb, A. Akella, K. Wang, G. Ricart, L. Landweber, C. Elliott, M. Zink, E. Cecchet, S. Kar, and P. Mishra, "The design and operation of cloudlab," in *Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference*, ser. USENIX ATC '19. USA: USENIX Association, Jul. 2019, pp. 1–14.

[29] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, A. Vahdat *et al.*, "Hedera: dynamic flow scheduling for data center networks." in *Nsdi*, vol. 10, no. 8. San Jose, USA, 2010, pp. 89–92.

[30] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center TCP (dctcp)," in *Proceedings of the ACM SIGCOMM 2010 Conference*, New Delhi, India, Aug. 2010, pp. 63–74. [Online]. Available: https://doi.org/10.1145/1851182.1851192

[31] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, Vaughan *et al.*, "The Llama 3 Herd of Models." arXiv, Nov. 2024, arXiv:2407.21783 [cs]. [Online]. Available: http://arxiv.org/abs/2407.21783