Journal of Network and Computer Applications 226 (2024) 103883

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Review article

Flow optimization strategies in data center networks: A survey ☆

Yong Liu [a], Tianyi Yu [a], Qian Meng [b,*], Quanze Liu [a]

[a] School of Information Science and Technology, Hangzhou Normal University, Hangzhou, 311121, China
[b] School of Mathematics, Hangzhou Normal University, Hangzhou, 311121, China

ARTICLE INFO

Keywords:
Flow optimization
Load balancing
Congestion control
Routing optimization
Flow scheduling

ABSTRACT

In the era of digitization, Data Center Networks (DCN) have emerged as a critical component supporting infrastructure for cloud computing, big data analytics, online services, and more. With the exponential growth in data transmission and processing demands, the issues related to the performance and efficiency of DCN have become increasingly urgent and complex. However, flow in DCN exhibits distinct characteristics compared to traditional networks, including uneven flow volume distribution, bursty transmission demands, and intricate network topologies. Conventional flow optimization strategies often fall short in this unique context. In recent years, researchers have actively proposed a plethora of innovative flow optimization strategies to address the challenges of enhancing the performance and efficiency of DCN. This survey aims to comprehensively review and analyze flow optimization strategies in DCN, encompassing various key areas of research and development dynamics such as load balancing, congestion control, routing optimization, flow scheduling, and flow security. Furthermore, it provides an in-depth discussion of the strengths and limitations of existing strategies to equip researchers with a comprehensive understanding and framework for this field. Moreover, this article explores future research prospects and trends to guide further innovative work in meeting the ever-growing demands of DCN.

1. Introduction

In the current digital era, Data Center Networks (DCN) have evolved into critical infrastructure that supports various technologies such as cloud computing, big data analytics, and online services (Noura et al., 2019; Qiu et al., 2018). With the continuous expansion of demands for large-scale data transmission and processing, the performance and efficiency of DCN have become exceptionally crucial. To consistently meet the demands of large-scale data, today's DCN typically comprises a multitude of hosts and switch devices, and their scale is experiencing exponential growth (Qiu et al., 2020). To support the ever-growing demands of network services, modern DCN employ specialized horizontal topologies such as Fat-tree (Al-Fares et al., 2008), VL2 (Greenberg et al., 2009), BCube (Guo et al., 2009), among others, to connect a vast number of commercial servers, enabling support for massive computational tasks. Meanwhile, in DCN, flow exhibits characteristics that are markedly different from traditional networks. Firstly, in DCN, approximately 80% of the flow volume consists of short flows, each less than 10 KB in size, despite these flows accounting for only 20% of the total data sent. However, the remaining 20% of the flow occupies 80% of the overall data sent. Secondly, flow in DCN is bursty in nature (Benson et al., 2010b). Even within an instantaneous time frame, multiple flows may arrive at network devices simultaneously, potentially leading to network congestion. In these network topologies, numerous path choices exist between pairs of end-hosts, making tasks such as load balancing, congestion control, routing optimization, flow scheduling and flow security of paramount importance. In this paper, we collectively refer to these measures as flow optimization strategies. Flow optimization strategies are crucial for the performance of DCN.

However, considering the distinctive topology and flow characteristics of DCN, some traditional flow optimization strategies do not perform well in the current DCN environment. Conventional load balancing algorithms like ECMP (Thaler and Hopps, 2000) typically perform hash calculations on multiple fields of packet headers and then map the hash value to available paths to achieve load balancing. Despite its relative simplicity in implementation, ECMP exhibits a range of issues, including hash collisions and the occurrence of elephant and mouse flow congestion, which can significantly degrade

☆ This work was supported by the National Natural Science Foundation of China under Grant 62302136, the Zhejiang Provincial Natural Science Foundation of China under Grant No. LQ22F010001, and the Starting Research Funds from the Hangzhou Normal University under Grant 4115C50221204137 and 4085C50220204093.

* Corresponding author.
E-mail addresses: yongliu@hznu.edu.cn (Y. Liu), 2022112011011@stu.hznu.edu.cn (T. Yu), mq@hznu.edu.cn (Q. Meng), 2022112011053@stu.hznu.edu.cn (Q. Liu).

https://doi.org/10.1016/j.jnca.2024.103883
Received 30 October 2023; Received in revised form 14 March 2024; Accepted 14 April 2024
Available online 21 April 2024

network performance. Traditional congestion control algorithms like DCTCP (Alizadeh et al., 2010) detect and respond to congestion issues by configuring Explicit Congestion Notification (ECN). However, in situations where network congestion is not severe, the DCTCP protocol can still potentially result in switch buffer overflows, subsequently triggering network congestion and packet loss problems. In DCN, without the ability to adjust routing strategies promptly based on the dynamic network conditions, network flow can lead to an uneven distribution of load among network nodes, resulting in resource waste and a decline in Quality of Service (QoS). Traditional routing optimization strategies, such as the shortest-path-first algorithm (Pióro et al., 2002), typically consider only the path selection for data flows while ignoring the network's load status and flow volume. This approach inevitably leads to network congestion issues, as it cannot adapt to the dynamically changing flow demands and network load conditions. In DCN, traditional flow scheduling strategies also exhibit certain shortcomings. Firstly, due to the complexity of flow models, some flow is sensitive to factors like latency, throughput, and deadlines, necessitating more refined adjustments in flow scheduling. Some flow may have knowledge of its data size, while others may not, posing additional challenges to flow scheduling. Secondly, the network model itself is highly intricate, with network bandwidth being dynamic. Available bandwidth may change by the time scheduling decisions are made, making it challenging to accurately capture real-time network conditions. In the environment of DCN, flow security is an important issue that cannot be ignored. As network attack methods continue to advance and diversify, traditional security mechanisms often fall short when faced with complex attack patterns. For example, Distributed Denial of Service (DDoS) attacks can easily circumvent defenses based on Access Control Lists (ACLs) by exploiting small flows (mouse flow) to launch large-scale attacks, leading to the exhaustion of network resources. Moreover, due to the enormous data traffic in DCN, traditional Intrusion Detection Systems (IDS) face a performance bottleneck when processing high-speed data streams, making it difficult to identify and respond to security threats in real time.

Therefore, over the past few years, researchers and engineers have proposed a plethora of mechanisms to address flow optimization issues in DCN. These mechanisms have varied focuses but primarily include load balancing, congestion control, routing optimization,flow scheduling, and flow security. Specifically, load balancing aims to achieve higher flow transmission efficiency, minimize end-to-end data transfer completion times, and enhance the overall utilization of network links. Congestion control involves adopting appropriate flow scheduling and management measures to prevent congestion occurrences, thereby mitigating adverse effects on DCN performance. Routing optimization aims to maximize data transfer speed, availability, and reliability while minimizing network congestion and latency. Flow scheduling, on the other hand, aims to ensure the efficient, reliable, and secure flow of data. The purpose of flow security is to ensure the security of data transmission within DCN, preventing malicious attacks and data breaches.

In this paper, we provide a comprehensive survey of flow optimization strategies within DCN. Fig. 1 illustrates the classification of our flow optimization strategy framework. Our research has made significant contributions in the following aspects:

- We conducted a comprehensive review of peer-reviewed literature published in high-impact journals and conferences over the past three years. These publications have received widespread acclaim and extensive citations within the domain of DCN, serving as invaluable references for research in this field. By categorizing these publications, we have furnished researchers with a structured framework for gaining a holistic understanding of the current state of leading research efforts.
- We conducted an in-depth exploration of various facets of flow optimization strategies, encompassing load balancing, congestion control, routing optimization, flow scheduling, and flow security solutions. Our analysis not only entails summarizing the merits of these approaches but also pinpointing their shortcomings. This endeavor aids researchers in gaining a more profound understanding of these strategies and in conducting more thorough evaluations.
- We presented future trends and expectations for the development of flow optimization strategies. This guidance is intended to assist researchers in achieving greater excellence in this field. Our review not only consolidates existing research findings but also provides insights into future research directions, thereby encouraging more innovative work.

The remaining content of this paper is structured as follows: Section 2 introduces essential background knowledge about DCN. Section 3 provides a detailed exploration of load balancing strategies. Section 4 delves deep into congestion control schemes. Section 5 explores routing optimization strategies. Section 6 comprehensively covers flow scheduling methods. Section 7 introduces the flow security scheme in detail. Section 8 presents future development trends and our expectations for flow optimization strategies. Finally, Section 9 concludes the paper.

2. Background

In this section, we commence with an extensive introduction to DCN. We start by elucidating the concept and essence of DCN, delving into the network's topology. Emphasis is then placed on the flow characteristics within DCN, including flow patterns, scales, and related aspects. Subsequently, we delve into SDN within DCN and its role in enhancing network manageability and performance. Lastly, we explore flow optimization.

2.1. DCN overview

During the early stages of computer network development, the concept of data centers was not yet well defined. The primary focus was on establishing connections between remote computers to facilitate remote data access and processing. These connections typically followed a point-to-point model and relied on traditional telecommunication technologies. However, with the widespread proliferation of the Internet and the explosive growth of digital information, leading enterprises such as Amazon, Microsoft, and Google Cloud began to construct increasingly extensive data centers (Di et al., 2013; Wang and Ng, 2010; D'Ambrosia et al., 2008). Typically, these data centers encompassed thousands of servers and necessitated high-performance network infrastructure to meet the demands for high throughput and low-latency data transmission. Large tech giants like Facebook, Google, and Microsoft, in support of their extensive online services and application ecosystems, established vast-scale DCN (Xia et al., 2016; Meza et al., 2018; Hammadi and Mhamdi, 2014). Today, in the context of rapid developments in cloud computing and big data, DCN have become a focal point in the technology landscape (Wang et al., 2015).

2.2. Topologies in DCN

To meet the substantial demands for flow and fulfill the high throughput and low-latency performance requirements of application flow, DCN topologies have been carefully designed. When using topological structures for packet forwarding, multiple paths of varying lengths are typically provided between servers (Bilal et al., 2012). In this section, we will introduce some representative topological structures.

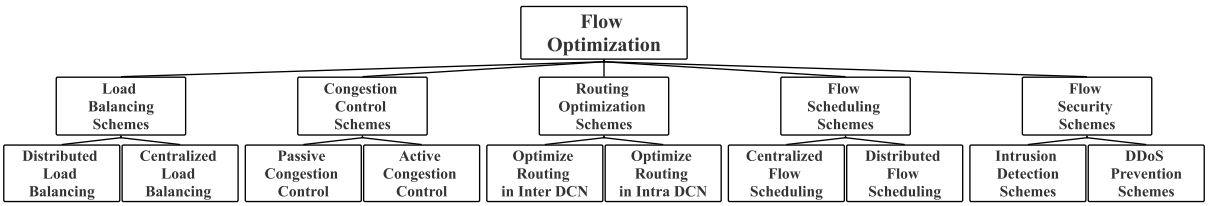


Fig. 1. Structure of this survey.

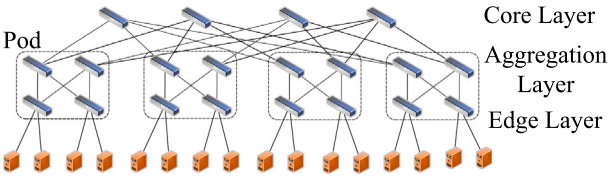


Fig. 2. Fat-tree topology.

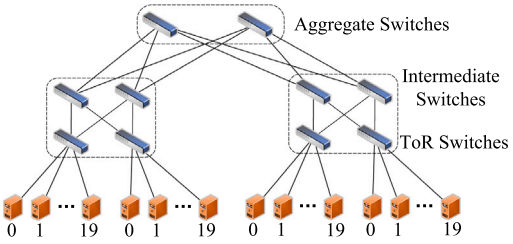


Fig. 3. VL2 topology for $N_A = 2$ and $N_I = 4$.

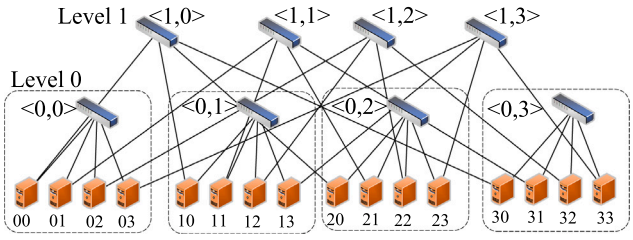


Fig. 4. BCube topology.

2.2.1. Fat-tree topology

The Fat-Tree (Al-Fares et al., 2008) topology is a structured topology designed around switches with the goal of maximizing end-to-end bandwidth while exhibiting excellent scalability. The Fat-Tree structure comprises three layers: the core layer, the aggregation layer, and the edge layer. In a k-pod Fat-Tree, each switch has k ports. The core layer is the top layer and consists of $\frac{k^2}{4}$ core switches. There are a total of k pods, each containing k switches. Both the aggregation and access layers consist of $\frac{k}{2}$ switches each. Each switch in the access layer can accommodate $\frac{k}{2}$ servers. Therefore, a k-pod Fat-Tree has k pods, with each pod accommodating $\frac{k^2}{4}$ servers. All pods together can accommodate $\frac{k^3}{4}$ servers, and there exist k paths between any two pods. In Fig. 2, we illustrate the Fat-Tree topology with $k = 4$.

2.2.2. VL2 topology

The VL2 (Greenberg et al., 2009) topology, proposed by Microsoft, was designed based on observations of flow characteristics in multiple real-world DCN. It employs a 2-tier network architecture using a Clos design and is primarily divided into two layers: the bottom server layer and the upper switch layer. Top of Rack (ToR) switches are used to connect these two layers. The switch layer comprises aggregate switches and intermediate switches, with each aggregate switch connecting to other aggregate switches through intermediate switches. In this design, if any one of the n intermediate switches fails, it only reduces the bidirectional bandwidth by $\frac{1}{n}$. This approach increases the number of end-to-end paths and enhances network robustness. Let the number of ports on an aggregate switch be N_A, and the number of ports on an intermediate switch be N_I. The uplink and downlink ports on an aggregate switch are $\frac{N_A}{2}$ each. The total number of ToR switches in the network is calculated as $N_I \times \frac{N_A}{4}$. Additionally, there are N_A aggregate switches and $\frac{N_A}{2}$ intermediate switches. The maximum supported number of servers in the network is $20 \times N_I \times \frac{N_A}{4}$. Fig. 3 illustrates the VL2 topology for $N_A = 2$ and $N_I = 4$.

2.2.3. BCube topology

BCube (Guo et al., 2009) is another novel network topology proposed by Microsoft researchers, designed for constructing large-scale DCN in a recursive manner. In the BCube structure, servers serve not only as places for data processing and storage but also as data forwarding entities, making it a hybrid topology. The construction of BCube follows these principles: Let k be the number of layers in the BCube network topology, and n be the number of ports on a switch. BCube_0 is the basic unit of the structure, composed of n servers and a switch with n ports. BCube_1 is constructed from n BCube_0 units and n switches with n ports each. Similarly, BCube_k is composed of n BCube_{k-1} units and n^k switches with n ports each. In BCube_k, servers have $k+1$ ports, numbered from 0 to k layers. It is important to note that in BCube, switches are only interconnected with servers. There are no interconnections between switches or between servers. Fig. 4 illustrates a n = 4 port switch interconnection in the BCube_1 topology, consisting of 4 BCube_0 units and 4 switches with 4 ports each.

2.3. Flow characteristics in DCN

In today's DCN, different data and applications exhibit distinct flow characteristics and requirements. First, consider real-time online services such as voice calls and video conferences. These applications are not only highly sensitive to latency but also demand a stable network connection to ensure high-quality communication. On the other hand, big data applications like data mining, analytics, and batch processing typically generate massive and continuous data flow (Benson et al., 2010b). Therefore, in this subsection, we primarily focus on these two types of flow.

(1) *Latency-sensitive flow:* Latency-sensitive flow, commonly referred to as "mouse flow" or "small flow" in DCN, represents a class of flow with extremely stringent latency requirements. The key characteristic of this type of flow is that its transmission speed must be fast enough to guarantee real-time performance and immediate responsiveness. To ensure the fulfillment of business requirements, latency-sensitive flow often needs to complete its transmission within a specified deadline.

(2) *Throughput-sensitive flow:* Throughput-sensitive flow, also widely referred to as "elephant flow" or "large flow" in DCN, represents a type of flow that places significant demands on network bandwidth. Unlike latency-sensitive flow, throughput-sensitive flow does not necessarily require real-time performance but prioritizes the efficiency of large-scale data transmission. Throughput-sensitive flow often necessitates

substantial bandwidth to support the transfer of extensive data volumes. These large data transfers can involve hundreds of megabytes or even more data, thus requiring DCN to provide ample bandwidth to accommodate these demands.

Therefore, the flow characteristics in DCN are typically a mixture of latency-sensitive flow and throughput-sensitive flow. These two types of flow are intertwined in DCN, collectively constituting the network's dynamic nature (Benson et al., 2010b,a).

2.4. Software-defined data center network

As cloud computing and big data continue to evolve, the scale and complexity of DCN are also increasing. Traditional DCN often face issues such as inflexible network topologies, high network latency, and inadequate security due to constraints in hardware resource allocation, network configuration, and maintenance. Data centers serve not only as places for data storage and processing but also host a variety of applications, including search engines, social networks, online games, video streaming, and more. These applications demand high network performance, including low latency, high bandwidth, and high reliability. To address these challenges, SDN technology has been widely adopted in DCN in recent years (Xia et al., 2014). As shown in Fig. 5, Software-Defined Data Center Networking refers to the application of SDN in DCN. It involves separating the control plane and data plane of the data center network and centralizing network control using a centralized controller, thereby achieving network flexibility, programmability, and automation. In this review, SDN technology has been extensively applied in various mechanisms and strategies to assist in optimizing flow management policies (Kreutz et al., 2014).

2.5. Problem of flow optimization

In the topology of DCN, there exist multiple potential paths between hosts, each with varying network conditions. Therefore, to achieve higher flow transmission efficiency and minimize end-to-end data transfer completion time, it is imperative to employ appropriate flow optimization strategies and techniques. The Multi-Commodity Flow problem (MCF) is a network flow problem where multiple commodities flow from different source nodes to different destination nodes in the network (Sen et al., 2013). Flow optimization in data centers is designed to address the issue of multi-commodity flows. We define the MCF problem as follows: Given a network topology denoted as $G = (V, E)$, where the bandwidth of edge $(u, v) \in E$ is given by $c(u, v)$. There are a total of k flows, denoted as $F_1, F_2, F_3, \ldots, F_K$, and each flow is defined as $F_i = (s_i, d_i, x_i)$, where s_i and d_i represent the source and destination nodes of flow F_i, and x_i is the demand. The flow through edge (u, v) for flow F_i is denoted as $F_i(u, v)$. The MCF problem is subject to the following constraints:

$$\text{Bandwidth constraint: } \sum_{i=1}^{K} F_i(u, v) \leq c(u, v). \tag{1}$$

$$\text{Demand constraint: } \sum_{w \in V} F_i(u, w) = x_i \Leftrightarrow \sum_{w \in V} F_i(w, u) = x_i. \tag{2}$$

Flow optimization alone cannot guarantee the optimal transmission strategy for the MCF problem because the optimal solution depends on flow demands. Therefore, flow optimization algorithms aim to minimize the maximum link utilization, expressed as:

$$\min \ max_{(u,v) \in E} \frac{\sum_{i=1}^{K} \sum_{s,d} F_i(s_u, d_v)}{c(u, v)}. \tag{3}$$

Solving this problem can provide insights into determining the optimal scheduling strategy. However, the MCF problem is NP-hard, and heuristic algorithms are commonly employed in existing flow optimization solutions. Therefore, in the following sections, we will delve into the solutions proposed by the academic community in recent years.

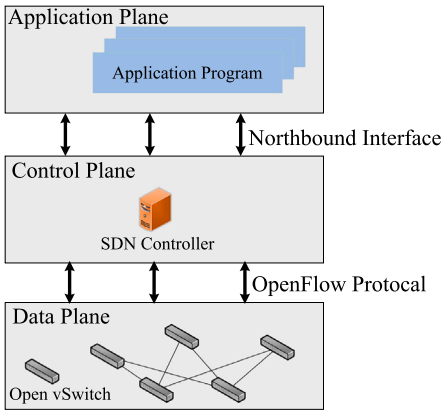


Fig. 5. SDN architecture.

3. Load balancing

In the context of DCN, the implementation of appropriate load balancing mechanisms is essential to enhance application performance and optimize the utilization of network resources. This section aims to provide a comprehensive overview of the load balancing problem in DCN, encompassing its overview, current research status, discussions, and insights.

3.1. Load balancing overview

In the current DCN, there exist multiple paths with equal costs between hosts, and different types of flow need to be transmitted. Therefore, a strategy is required to enhance application performance and effectively utilize network resources. Load balancing mechanisms in DCN are a technology used to distribute network flow and requests to ensure the efficient utilization of network resources, improve performance, and enhance availability (Zhang et al., 2018).

In the context of traditional ECMP (Thaler and Hopps, 2000), a mathematical model can be established to optimize the load balancing of flow across equivalent links. Given a network topology represented by the graph $G(V, E)$, where the set V includes network nodes, typically consisting of switches and hosts, and the set E represents the collection of links, the optimization objective of ECMP is to minimize the maximum link utilization (Sen et al., 2013), defined as:

$$\min(\max_{e \in E} \frac{y_e}{C_e}). \tag{4}$$

Here, e represents a unidirectional link in the network topology G, and $e \in E$. y_e represents the flow load on link e, and C_e represents the capacity of link e.

When designing load balancing algorithms, it is essential to consider several key objectives comprehensively to ensure the efficient operation of the network. We summarize several main objectives that load balancing algorithms need to take into account: **(1) The Impact of Load Balancing Algorithms on Flow Completion Time (FCT):** How do different load balancing algorithms affect flow completion time under various network loads and flow patterns? **(2) Dynamic Adaptation to Network Conditions:** How do load balancing algorithms dynamically adjust based on real-time network states (such as link utilization, changes in flow patterns, etc.) to optimize network throughput? **(3) Cost-Efficiency Analysis:** How to balance the relationship between equipment costs (including switches and host resources such as CPU, memory) and network performance when implementing load balancing strategies?

Table 1
Flow-base & Packet-base & Flowlet-base Load Balancing Schemes.

Ref	Year	Granularity	Evaluation method	Advantages	Disadvantages
ECMP (Thaler and Hopps, 2000)	2000	Flow	Real network environment	Enhancing network bandwidth and fault tolerance	Unequal flow distribution and neglecting path bandwidth differences
VL2 (Greenberg et al., 2009)	2009	Flow	Real network equipment with simulated flow testing	Achieves network congestion-free and provides unified high bandwidth	Directory services may require further optimization for extremely large-scale scenarios
PLB (Qureshi et al., 2022)	2022	Flow	Custom simulator and real networks	Effectively resolves ECMP hash collision issues	Limiting scalability, IPv6 tag usage not widely supported
Mohammed and Tăpuş (2023)	2023	Flow	Custom simulation tool with real workloads	By analyzing big data, effective usage patterns can be captured to achieve energy optimization	The effectiveness needs to be verified in larger-scale scenarios due to the limited scale of the experiment.
RPS (Dixit et al., 2013)	2013	Packet	Real network equipment with simulated flow testing	Leveraging multiple path resources more efficiently	May result in packet reordering
QDAPS (Huang et al., 2021)	2021	Packet	NS2 and Mininet topology simulation	Effectively resolves packet reordering issues	High computation complexity on switches
RMC (Zou et al., 2021a)	2021	Packet	NS2 and Mininet topology simulation	Effectively resolves packet reordering in asymmetric topologies	High computation complexity on switches
APS (Liu et al., 2022a)	2022	Packet	NS2 large-scale topology simulation, real network test-bed evaluation	Effective long and short flow conflict resolution using path isolation	Challenging in distinguishing long and short flow issues
Letflow (Vanini et al., 2017)	2017	Flowlet	OMent++ simulation with real workloads	Achieving simplicity	Weak responsiveness may lead to short-term congestion
ILB (Zhang et al., 2022)	2022	Flow & Flowlet	NS2 topology simulation, real network Workload	Effective separation of short and long flows on transmission paths	Flow type identification may have errors, difficult to precisely differentiate specific flow sizes
PLB (Liu et al., 2020)	2020	Flowlet	NS3 topology simulation, real network Workload	Efficient network resource detection	Does not address packet reordering
DLB (Zhang et al., 2023)	2023	Flow & Flowlet	NS2 topology simulation, real network workload	First load balancing solution considering flow transmission deadlines	Practical application challenges, requires modifications to host and switch

3.2. Load balancing schemes

Currently, the main load balancing solutions can be broadly categorized into two types: distributed and centralized. In the following sections, we will provide detailed insights into these two types of load balancing schemes.

3.2.1. Distributed load balancing

Distributed load balancing primarily addresses the issue of distributing flow among multiple servers, nodes, or data centers. By evenly distributing flow, it helps avoid the uneven utilization of server resources, thereby improving the overall system's performance and availability. Generally, distributed load balancing solutions are typically deployed and implemented on critical devices within the data center, including hosts and network switches.

Switch-based scheduling schemes can be categorized into four types: packet-based, flow-based, flowlet-based, and mixed granularity-based. In the following sections, we will provide a detailed introduction to these four types of schemes. Table 1 summarizes the load balancing scheme based on flow, packet, and flowlet, and Table 2 summarizes the load balancing scheme based on mixed granularity.

(A) Flow-based load balancing scheme

ECMP performs well when flow sizes are equal. When a large amount of flow arrives simultaneously, hash collisions may occur. To address these issues, a scheme called PLB (Qureshi et al., 2022) was proposed as an improvement over ECMP. PLB builds upon the ECMP foundation but introduces IPv6 tagging for hash calculations to alleviate hash collision problems. Furthermore, PLB incorporates a mechanism for detecting congestion signals. Once congestion is detected, it rehashes flows and redirects flow to non-congested paths, providing a response to link failures and recovery. PLB is deployed on all Google data center servers, supporting TCP and Pony Express transport protocols, covering nearly all internal transmission needs. It implements a fast, incremental deployment method that allows the sender to enable flow-based adjustments unilaterally without interrupting the flow, thus enhancing network performance without interference. PLB demonstrates significant performance improvements in several aspects, including reducing link load imbalance, lowering packet loss, reducing RPC tail latency and the occurrence of missed deadlines, with minimal increase in CPU usage or retransmission penalties.

The VL2 architecture (Greenberg et al., 2009) by Greenberg et al. achieves efficient connectivity and performance isolation in data center networks through the use of a flat address space, Valiant Load Balancing (VLB), and a directory service. By adopting a Clos topology, it optimizes cost and bandwidth allocation, while Valiant technology provides immunity to flow hotspots, avoiding performance bottlenecks. Address separation and directory services enhance dynamic scheduling and scalability.

M. Mohammed et al. developed a method (Mohammed and Tăpuş, 2023) to reduce the energy consumption of mobile cloud computing by combining big data analytics to predict resource demand and dynamically adjust cloud resource allocation. Through techniques such as clustering, classification, regression, and time series analysis, the method intelligently analyzes usage data to achieve efficient resource management. This strategy significantly enhances energy efficiency by reducing over-provisioning and improving utilization rates.

(B) Packet-based load balancing scheme

While flow-based load balancing methods can effectively address packet reordering issues, they may sometimes lead to suboptimal link utilization. Therefore, researchers have proposed several packet-based load balancing schemes to tackle these challenges.

RPS (Dixit et al., 2013) is a simple packet-based load balancing scheme. On switches, RPS divides each flow into packets and randomly

Table 2
Base-mixed granularity load balancing schemes.

Ref	Year	Granularity	Evaluation method	Advantages	Disadvantages
FAMG (Lu et al., 2023b)	2023	Flowlet & Packet	NS3 topology simulation, real network workload	Hybrid approach combining per-flowlet and per-packet granularity	Sacrificing some large flow completion times to improve small flow performance
BOBBLE (Xu et al., 2021)	2021	Flowlet & Packet	NS3 topology simulation, real network Workload	Effective scheduling of long and short flows at different granularities	Challenging in distinguishing long and short flows
TLB (Hu et al., 2021)	2021	Adaptive Granularity	NS2 topology simulation, real network workload	Efficiently adjusts forwarding granularity based on characteristics of short flows	High computation complexity on switches
LBT (Wang et al., 2023)	2023	Adaptive Granularity	NS2 topology simulation, real network workload	Routing different types of flow with varying granularity	High computation complexity on switches
AG (Liu et al., 2023b)	2023	Adaptive Granularity	NS2 topology simulation, real network workload	Dynamically adjusts granularity based on path differences	Does not consider long and short flow congestion

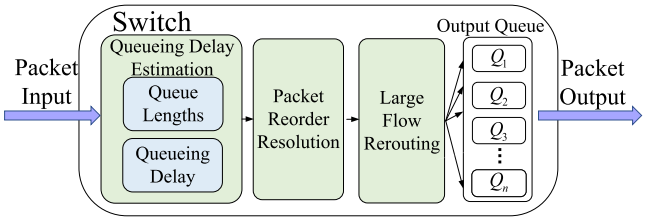


Fig. 6. Workflow of QDAPS.

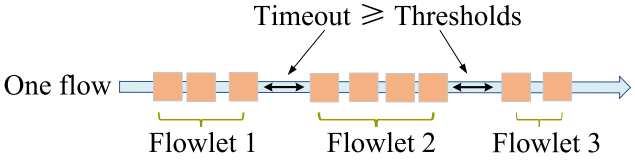


Fig. 7. Principle of flowlets.

sprays these packets across all available paths to fully utilize the bandwidth of parallel paths. However, because the states on different paths may vary, this can lead to packets arriving in different orders, resulting in significant packet reordering issues that can impact flow completion time and throughput.

To address the issue of packet reordering, the QDAPS (Huang et al., 2021) proposes a packet-level load balancing mechanism that is sensitive to queueing delays. Specifically, QDAPS selects packet paths based on the queuing delay in the output port buffer of the previous packet. This ensures that packets arriving earlier are transmitted before those arriving later, effectively resolving the reordering issue. Fig. 6 illustrates the workflow of QDAPS.

RPS could potentially lead to severe packet reordering issues in cases of asymmetric or non-uniform network topologies. To address this concern, researchers have introduced a packet-level load balancing scheme known as RMC (Zou et al., 2021a). RMC employs an assessment mechanism on switches that evaluates the likelihood of causing packet reordering based on local queue lengths and global path delays. Additionally, RMC incorporates a redundancy-optimized coding technique aimed at mitigating the impact of network asymmetry on packet reordering issues.

Researchers have introduced a scheduling scheme known as APS (Liu et al., 2022a). The APS scheme dynamically separates mouse flows from elephant flows across multiple available paths based on the bursty nature of mouse flows, thereby achieving flow isolation. Specifically, APS is deployed on switches and adjusts the transmission paths of flow by computing the minimum number of paths required to guarantee latency limits for mouse flows. When mice flows are in a closed state, APS utilizes RPS (Dixit et al., 2013) to transmit elephant flow packets across all available paths, thus improving the throughput of elephant flows. However, when mice flows are in an open state, APS employs the ECMP (Thaler and Hopps, 2000) strategy, sending elephant flow packets only on paths that do not include mice flows to avoid unnecessary interference with mice flows. APS enables a more precise evaluation of its effectiveness and efficiency in managing heterogeneous traffic by comparing flow distributions under different workloads such as data mining, web servers, and Hadoop.

(C) Flowlet-based load balancing

To address load imbalance problems in DCN, a scheduling unit based on flow characteristics, known as flowlet (Kandula et al., 2007), has been introduced. This technology divides flows into smaller sections known as flowlet. A new flowlet is recognized only when the time interval between packets in a flow exceeds a specified flowlet timeout. In Fig. 7, we depict the principle of flowlets, where a new flowlet is created when the time gap between packets exceeds the specified thresholds.

The LetFlow (Vanini et al., 2017) algorithm optimizes data center network flow allocation through a flowlet-based strategy to meet the high-capacity demands of bipartite networks. Unlike traditional ECMP, LetFlow achieves load balancing by random path selection and dynamically adjusting the size of flowlets, without relying on path status. This flexibility allows it to adapt to network changes, improving resource utilization and reducing congestion.

ILB (Zhang et al., 2022) proposes a load balancing mechanism that relies on flow type identification and path segregation. ILB categorizes flows into three distinct classes: tiny flows, small flows, and long flows. It devises distinct load balancing strategies tailored to each flow category. Specifically, tiny flows are subject to flow-level load balancing, while small and long flows are managed at the flowlet level. Simultaneously, ILB takes into consideration the real-time assessment of path conditions to dynamically opt for the output queue path with the lowest load.

PLB (Liu et al., 2020) adopts a flowlet-based routing approach coupled with adaptive path probing strategies, selective flow rerouting, and precise congestion metrics to achieve load balancing in DCN. Simultaneously, PLB employs adaptive path probing strategies that dynamically adjust the number of probing paths. This allows for increased precision in path probing when needed while reducing unnecessary path probing in cases of lighter network loads, thereby minimizing bandwidth overhead.

Furthermore, existing data center load balancing solutions often lack the ability to perceive the transmission deadline requirements of flows and cannot intelligently redirect flow based on their urgency. Therefore, a load balancing scheme based on deadline-driven flows DLB (Zhang et al., 2023) has been proposed. DLB can sense the transmission deadline requirements of flows and intelligently redirect flow according to their urgency to maximize meeting their transmission time requirements.

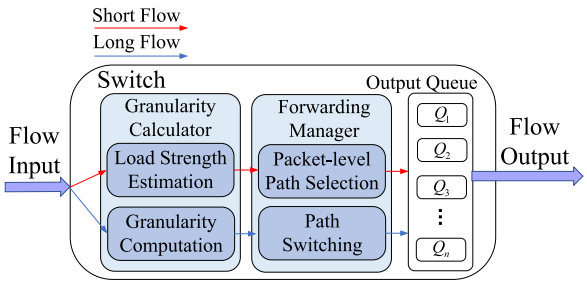


Fig. 8. TLB overview.

(D) Mixed granularity-based load balancing

Traditional load balancing solutions in DCN often employ static routing granularities, such as per-flow or per-packet granularity. These methods use fixed routing granularities and do not fully consider the dynamic nature of flow characteristics and network topology. To overcome these issues, some research efforts have proposed load balancing schemes based on mixed granularities. These approaches aim to achieve more flexible and efficient network performance optimization by dynamically adapting to changing network conditions and flow characteristics.

In this context, the FAMG (Lu et al., 2023b) method introduces an innovative load balancing strategy with the core concept of dynamically selecting the appropriate routing granularity based on the characteristics of flows. Specifically, the FAMG method classifies flow into two main categories, namely, large flows and small flows, by identifying their sizes. For large flows, FAMG employs per-flowlet granularity for routing to fully utilize network bandwidth and prevent packet reordering issues. On the other hand, for small flows, per-packet granularity is utilized to enhance link utilization.

The BOBBLE (Xu et al., 2021) represents another mixed-granularity load balancing strategy, with its core concept being the division of flow into large flows and small flows, each receiving the most suitable routing granularity. The specific operations involve flow classification at the ingress switches and the addition of corresponding labels. For large flows, BOBBLE employs a flowlet-based random scheduling algorithm, splitting large flows into multiple flowlets for load balancing to maximize network bandwidth utilization.

The TLB (Hu et al., 2021) method focuses on addressing congestion issues related to both small and large flows. The TLB method adaptively adjusts the switching granularity of long flows based on the load conditions of short flows to allocate network resources efficiently. When the load on short flows is high, TLB increases the switching granularity of long flows, thereby reserving more low-congestion paths for short flows, reducing their queuing delays. Conversely, when the load on short flows is low, TLB decreases the switching granularity of long flows to enhance link utilization. In Fig. 8, we illustrate the working principle of TLB. The TLB scheme effectively enhances the overall experience of end-users by simultaneously reducing the completion time of short flows and increasing the throughput of long flows. For mixed flow patterns such as web searches and data mining, TLB achieves a balance between short flow response times and long flow data transfer efficiency, offering a solution that is both efficient and fair in terms of service quality.

In addition to the aforementioned methods, the LBT (Wang et al., 2023) approach focuses on load balancing for heterogeneous flow in DCN. This method classifies flow into three categories, including short flows, long flows, and Best-Effort (BE) flows, and dynamically adjusts forwarding granularity based on their load conditions and link congestion to achieve load balancing for heterogeneous flow.

Lastly, the AG (Liu et al., 2023b) method addresses packet reordering issues in networks with asymmetric topologies. AG offers a solution by dynamically adjusting the size of Packet Trains based on the network's asymmetry level to reduce reordering problems. In particular, AG performs load balancing on a Packet Train basis and adaptively modifies the size of Packet Trains based on the measured path asymmetry. When there is significant variation in network paths, AG increases the size of Packet Trains to reduce the occurrence of packet reordering. Conversely, when network path differences are minimal, AG decreases the size of Packet Trains to enhance link utilization.

3.2.2. Centralized load balancing

The core concept of centralized load balancing is to coordinate and manage all flow distribution and decision-making through a controller. The controller can monitor the overall state of the network or system, including the load on servers, nodes, or data centers. In Table 3, we enumerate centralized load balancing strategies employed in recent years.

Weighted multipath approaches based on SDN have gained widespread attention. These methods intelligently distribute flow based on the current state of the network. However, this approach is constrained by the flow table capacity of switches. Expanding flow table capacity often requires costly hardware support, making it challenging to achieve ideal weighted ratios for each flow group. Furthermore, existing flow distribution methods like WCMP (Zhou et al., 2014) and Niagara (Kang et al., 2015) primarily optimize to minimize port oversubscription or flow imbalance without sufficiently considering the optimization of transmission times for different flow groups. NAMP (Cheng and Jia, 2020) is a network-aware approach that optimizes the allocation of flow table rules through central scheduling to minimize transmission times for different flow groups.

LBPP (Liu et al., 2023a) aims to achieve real-time responsiveness to dynamic network conditions and load balancing with low decision latency and overhead. This method implements load balancing at the packet level. Specifically, LBPP introduces a network modeling approach based on Graph Neural Networks (GNN) (Wu et al., 2020), referred to as GNN-Behavior. By learning the intrinsic relationships between fundamental network behaviors, this approach comprehensively models information related to network topology, flow control, and more. In Fig. 9, we present the main architecture of LBPP.

Distributed routing algorithms like BCube Source Routing (BSR) can lead to severe flow conflicts in worst-case scenarios, resulting in bandwidth utilization falling below 50%. To address these challenges, CDPFS (Chung et al., 2023) proposes two controller-based dynamic parallel flow scheduling algorithms: CDPFS and CDPFSMP. The aim is to reduce flow conflicts and improve load balancing in the BCube cloud data center network. Specifically, these algorithms employ a controller to establish a global view and allocate flow paths to minimize potential flow conflicts. They construct independent path sets for each data flow, namely the SP (Single-Path) and NSP (Non-Single-Path) graphs, and use parallel computation to distribute flow paths.

There may be conflicting objectives between users and providers in load balancing problems. For example, users may prioritize low latency, while providers are concerned with reducing energy consumption. Therefore, an effective load balancing algorithm should make decisions based on a comprehensive consideration of factors such as user experience, provider efficiency, and fairness. To address this challenge, a load balancing solution based on cooperative game theory called LEWIS (Kishor et al., 2023) is proposed. LEWIS considers both latency and energy consumption as two objectives and seeks the system's optimization solution by computing the Nash equilibrium point of the game.

3.3. Discussion and insights on load balancing

In this section, we delve into the relevant topics of distributed load balancing and centralized load balancing, providing in-depth insights. Firstly, the primary goal of distributed load balancing is to

Table 3
Centralized load balancing schemes.

Ref	Year	Granularity	Evaluation method	Advantages	Disadvantages
NAMP (Cheng and Jia, 2020)	2020	Flow	NS3 topology simulation with RYU controller	Addressing the limited flow table issue on switches	Controller computation cycle may impact strategy performance
LBPP (Liu et al., 2023a)	2023	Packet	Mininet topology simulation with Ryu controller, real network workload	Predictive load scheduling achieved through control and data plane collaboration	May introduce packet reordering issues
CDPFS (Chung et al., 2023)	2023	Flow	Mininet topology simulation with Ryu controller, simulated flow testing	Achieving load balancing allocation for single-path and multi-path flows	Application effects on other topology structures not widely tested
LEWIS (Kishor et al., 2023)	2023	Flow	Custom simulation tool, real network workload	Concise algorithm design, easy implementation and applicability	Experimental parameter settings overlook network topology influence

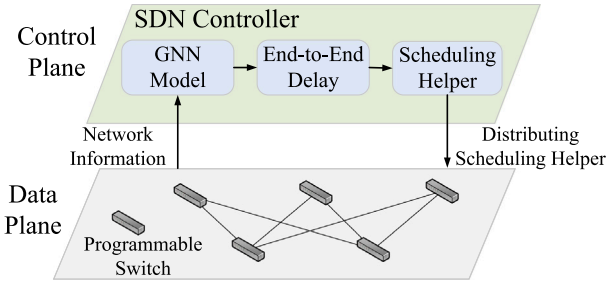


Fig. 9. The main architecture of LBPP.

evenly distribute flow across multiple servers, nodes, or data centers to avoid overloading certain links, thereby enhancing the overall system performance. Subsequently, based on the granularity of scheduling, distributed load balancing can be categorized into several methods, including those based on different granularities, such as flow-based, packet-based, flowlet-based, and mixed granularity-based. Each granularity method has its own unique advantages and limitations. Flow-based ECMP methods can effectively address packet reordering issues but may lead to uneven flow distribution. Packet-based RPS methods can improve link utilization but may introduce severe packet reordering. Flowlet-based methods, by segmenting flows, partially balance link utilization and packet reordering issues. Mixed granularities offer greater flexibility, allowing the choice of scheduling granularity based on flow type. In different network environments and flow characteristics, selecting an appropriate load balancing strategy is crucial, and the dynamic adjustment of strategies becomes particularly important.

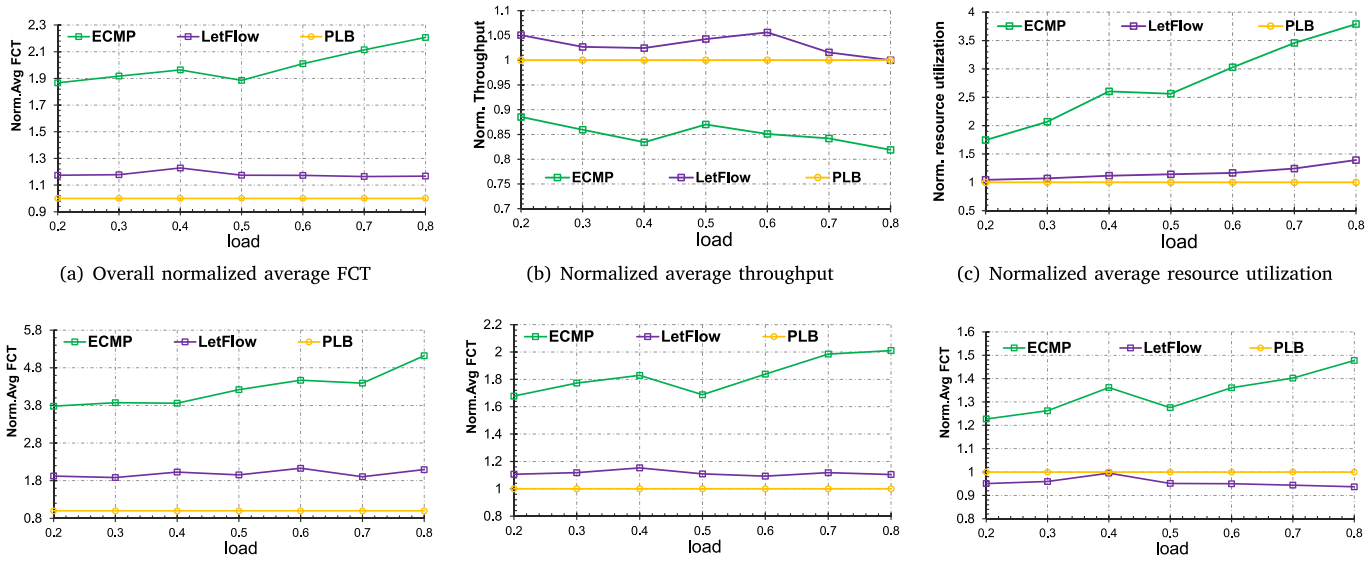
Although distributed load balancing excels in effectively balancing loads, load balancing mechanisms based on switches typically require customized switch chips or the use of programmable switches, posing certain challenges in large-scale deployments. Moreover, with distributed load balancing mechanisms, individual hosts or switches may struggle to fully comprehend the overall network environment, making it challenging to make optimal load balancing decisions. Therefore, centralized load balancing strategies, which leverage a global perspective to schedule flow in DCN, can achieve better load balancing. Firstly, centralized load balancing allows for a global view of flow scheduling, enabling a more comprehensive understanding of the entire network environment. This enables the more precise development of flow allocation strategies for optimal performance. Additionally, centralized load balancing typically exhibits greater adaptability, better addressing dynamic network changes, as it can comprehensively monitor and adjust the entire network state. However, centralized load balancing solutions also face several challenges that require further research. Firstly, achieving coordination between the control plane and data plane is a crucial challenge, necessitating the design of low-latency and efficient interoperability mechanisms. Secondly, controller performance may become a bottleneck, leading to the need for methods like distributed controllers to enhance scalability. Finally, the accuracy of flow prediction directly impacts scheduling quality, necessitating continuous improvements in network behavior modeling and flow prediction techniques.

To facilitate a more detailed discussion on load balancing, we have conducted corresponding quantitative experiments. The simulation of the data plane was conducted using the Mininet (Dholakiya et al., 2021) network simulation software. In the control plane simulation, we utilized the Ryu controller (Li et al., 2020), leveraging the OpenFlow protocol (Pfaff et al., 2012). The experimental topology was designed based on a 6-pod Fat-tree architecture. In our simulation experiment, we selected the web search (Alizadeh et al., 2010) workload widely used in real data centers. Dynamic flow was generated using existing client/server applications (Bai et al., 2016). In this study, we primarily focused on three performance metrics: FCT, throughput, and CPU utilization.

In this study, we replicated and compared three load balancing schemes: ECMP (Thaler and Hopps, 2000), PLB (Qureshi et al., 2022), and Letflow (Vanini et al., 2017), aiming to explore their fundamental concepts and performance outcomes. To better compare and observe the performance of different schemes, we normalized the test values of each scheme to the PLB scheme. First, Fig. 10(a) illustrates the overall average Flow Completion Time (FCT) under the web search workload for the three schemes. Specifically, the Letflow scheme, which controls flow based on finer-grained flowlets as opposed to ECMP's traditional hashing approach, effectively avoids hash collision issues. The innovation of the PLB scheme lies in its introduction of a flow label on top of the traditional four-tuple hash, achieving a five-tuple hash match. This improvement significantly enhances the resolution of hash collisions for large and small flows. According to the data in Fig. 10(e), the PLB scheme reduces the average FCT by 14.81% to 18.57% compared to the Letflow scheme, with an even greater reduction in FCT for small flows of 47.98% to 53.04%. This is because using flowlet granularity for transmission can cause packet reordering issues. However, as shown in Fig. 10(b), Letflow slightly outperforms PLB in terms of throughput, with an average throughput increase of 2.52% to 5.73%. This is mainly attributed to Letflow's fine-grained flowlet-based flow control strategy, which more effectively utilizes multipath for packet transmission, thereby enhancing the overall network throughput. Lastly, as depicted in Fig. 10(c), under high network load conditions, the ECMP scheme faces significant challenges with large and small flow collisions and packet reordering issues, leading to a noticeable increase in CPU utilization compared to PLB and Letflow. This phenomenon indicates that under high load conditions, ECMP requires more resources to address flow conflicts and reordering issues, resulting in higher CPU utilization.

4. Congestion control

In a DCN, numerous factors can lead to network congestion, emphasizing the need for the development of an effective congestion control scheme. This section begins by defining congestion control schemes



(a) Overall normalized average FCT (b) Normalized average throughput (c) Normalized average resource utilization

(d) Mice flow normalized average FCT (0, 100 KB) (e) Medium flow normalized average FCT (100 KB, 10 MB) (f) Elephant flow normalized average FCT (10 MB, +∞)

Fig. 10. Performance comparison of web search workload on load balancing schemes.

and introducing the primary optimization objectives for such schemes. Subsequently, it provides an overview of some outstanding congestion control schemes that have emerged in recent years. Finally, the section offers insights into this topic.

4.1. Congestion control overview

DCN serve as the core for large-scale computation and data processing, characterized by their highly intricate topological structures and substantial data flow. Effective congestion control protocols are paramount in such networks. Network congestion can occur when the current incoming flow exceeds the capacity of the links. Consequently, the primary objective of congestion control is to prevent congestion by appropriately managing and scheduling flow, thus alleviating the adverse effects of congestion on performance (Huang et al., 2018).

Therefore, one of the foremost goals of congestion control protocols is to prevent congestion points within the network. These congestion points can potentially result in packet loss, increased latency, and a decline in network performance. We summarize several key objectives that congestion control algorithms need to consider: **(1) The Impact of Congestion Control Protocols on Small and Large Flow FCT:** How do different congestion control protocols affect the FCT of small and large flows, especially under varying network load conditions? **(2) Congestion Control and Deadline-Driven Flow Management:** How do effective congestion control mechanisms decrease the proportion of flows that fail to meet their deadlines? **(3) Congestion Control under Dynamic Network Load:** How do congestion control algorithms adapt to the dynamic changes in network load to optimize overall throughput?

4.2. Congestion control schemes

Typically, congestion control can be categorized into two main types: active congestion control and passive congestion control (Liu et al., 2013; Rojas-Cessa et al., 2015; Noormohammadpour and Raghavendra, 2017). In Tables 4 and 5 below, we provide a detailed list of recent congestion control schemes.

4.2.1. Passive congestion control

Passive congestion control algorithms are designed to manage congestion within a network by automatically detecting and responding to congestion situations without requiring active notifications or requests for data senders to reduce their transmission rates. These algorithms rely on monitoring network flow, queue lengths, and other metrics while employing adaptive measures to maintain network performance and reliability.

In the context of the traditional congestion control scheme, DCTCP, the utilization of the ECN mechanism is pivotal. When network switches within a DCN detect congestion, they apply ECN labels to transmitted data packets. Upon arrival at the receiving end, if the ECN counter value at the receiver exceeds a specific threshold, DCTCP recognizes network congestion and subsequently reduces the data transmission rate to mitigate the worsening of the congestion. The following mathematical modeling represents key aspects of DCTCP: First, calculation of the estimated fraction of marked data packets at the sender:

$$\alpha = (1 - c) \times \alpha + c \times F. \tag{5}$$

Here, α represents the sender's estimate of the fraction of marked data packets. F is the fraction of marked data packets in the last data window, and c is the weight assigned to past estimates when calculating α. When α is close to 0, it indicates lower congestion, and when α is close to 1, it suggests higher congestion. Second, adjustment of the congestion window size:

$$cwnd = cwnd \times (1 - \alpha/2). \tag{6}$$

The congestion window size, denoted by cwnd, decreases as α approaches 1, signifying congestion in the network. Conversely, when α is closer to 0 (indicating no congestion), the reduction in window size is smaller.

The passive congestion control algorithm is described in Table 4. Traditional TCP protocols typically select only one path for data transmission. When congestion occurs on a chosen path, other paths may still have available bandwidth resources. Therefore, there is a need for a mechanism to achieve dynamic flow load balancing in DCN to enhance performance. In response to this, a new TCP protocol was called MA-TCP (Xia et al., 2021). MA-TCP is designed to be aware of the path migration of TCP flows, thereby avoiding the reduction of congestion window size during flow migration due to packet reordering. Specifically, MA-TCP employs a random path probing strategy to

Table 4
Passive congestion control schemes.

Ref.	Year	Congestion signal	Congestion aware	Modification Place	Evaluation method	Advantages	Disadvantages
DCTCP (Alizadeh et al., 2010)	2010	ECN	Yes	Switch	Simulation based on NS3,with real network workloads	The algorithm implementation is simple	The convergence speed is slower compared to TCP
MA-TCP (Xia et al., 2021)	2021	RTT & ECN	Yes	Host	Topology simulation based on NS3, with real network workloads	Perceiving path migration and avoiding rate decreases during flow migration	Does not consider long and short flows issue
BBC (Bai et al., 2021)	2020	ECN	Yes	Switch	Topology simulation based on NS2, with real network workloads	Solving congestion issues caused by minimal buffer size	Does not consider long and short flows issue
Cloudburst (Zeng et al., 2022)	2022	None	No	Host	Topology simulation based on NS2, tested by simulating flow	Simple deployment with minor modifications on hosts	Redundant packets increase network load
T-RACKs (Abdelmoniem and Bensaou, 2021)	2021	None	No	Host	Topology simulation based on physical network equipment	Addressing the problem of long TCP timeouts	Scalability of this mechanism not verified in large-scale data centers
A^3DCT (Lu et al., 2023a)	2023	ECN	Yes	Host	Topology simulation based on NS3, with real network workloads	Prioritizing the handling of residual flows with shorter completion times	Algorithm performance depends on parameter settings
DCQUIC (Tan et al., 2021)	2021	None	Yes	Host	Topology experiments in a private DCN, tested by simulating flow	Introducing QUIC in DCN	Compatibility and deployment complexity with mainstream protocols
AP (Zou et al., 2021b)	2020	None	No	Host	Topology simulation based on NS2, tested by simulating flow	Effectively mitigates TCP Incast issues	Balancing Incast optimization with throughput loss is challenging
Flash (Gao et al., 2023)	2023	ECN	Yes	Host & Switch	Topology simulation based on NS2, with real network workloads	Considering both flow scheduling and congestion control	Does not consider performance under more complex network topologies

Table 5
Active congestion control schemes.

Ref.	Year	Congestion Signal	Congestion aware	Modification Place	Evaluation method	Advantages	Disadvantages
D^3 (Wilson et al., 2011)	2011	None	Yes	Host & Switch	Real network equipment with simulated flow testing	Provides improved support for flow concurrency	Requires application layer support for flow size and deadline information
RSCAT (Diel et al., 2022)	2022	None	Yes	Host	Utilizing Ryu as a controller with simulated flow	Adaptable to optimal adjustment through reinforcement learning	Requiring strong server support for handling large-scale networks
RoCC (Menikkumbura et al., 2023)	2023	None	Yes	Host & Switch	Topology simulation based on OMNeT++, with real network workloads	Achieving rapid response by actively calculating the fairness rate	Requiring some host and switch adaptations to support RoCC functionality
Swift (Kumar et al., 2020)	2020	Delay	Yes	None	Assessed using a homemade simulation tool	Simple design, relies only on host latency measurements	Delay signal itself may have a low information content compared to ECN
PPO (Ketabi et al., 2023)	2023	Delay	Yes	Host	Topology simulation based on Mininet, and simulates communication with simulated flow	Dynamically optimize protocol adaptation to network changes	Chosen reward function and environment simulation may not fully match reality
FACC (Chen et al., 2023)	2023	Delay	Yes	Host	Using OMNeT++ for topology simulation and real network workloads	Better meeting diverse flow requirements	Does not incorporate other flow characteristics into decision-making

detect congestion states on available paths and reroute congested flows to paths with lower congestion levels.

With the rapid growth in link rates within DCN, there has been a significant discrepancy as the size of switch buffers has not kept pace with this expansion. This has resulted in a substantial reduction in available buffer space. To address this challenge, the novel approach of BCC (Bai et al., 2021) has been introduced. BCC introduces an entirely new buffer-aware active queue management mechanism that can be seamlessly implemented by adding shared buffers to existing switches. A testbed was constructed using 6 servers connected to an Arista 7060CX-32S 100Gbps switch equipped with Broadcom Tomahawk chips. This switch features 4 Memory Management Units (MMUs), each equipped with a 4 MB buffer, and dynamic buffer allocation is completed within a single MMU. The results demonstrated that BCC can not only be easily deployed on commercial switches but also proves effective under various MMU configurations, confirming its viability and efficiency on actual commercial hardware.

In DCN, there exist small flows generated by interactive applications, which often result in a multitude of small TCP flows. Due

to network topology and suboptimal TCP configurations, these small flows are susceptible to congestion-induced packet loss. To address this issue, a solution named Cloudburst (Zeng et al., 2022) has been proposed. Cloudburst does not rely on congestion awareness, ECN, or RTT strategies. Instead, it achieves fast and reliable transmission of small flows by parallelizing the transmission of redundant coded packets using Forward Error Correction (FEC) encoding across multiple paths.

Furthermore, T-RACKs (Abdelmoniem and Bensaou, 2021) also focus on addressing the problem of packet loss in small flows. Its rapid TCP recovery mechanism proactively triggers TCP's fast retransmission for small flows at the emulation layer between virtual machines and network hardware. This approach successfully reduces the transmission time of small flows without the need for modifications to the TCP protocol stack.

A^3DCT (Lu et al., 2023a) also addresses the requirements of small flows. This approach employs the Cubic Acceleration TCP algorithm to enhance the completion time of short flows. It utilizes the Shortest Remaining Flow Time (SRFT) strategy to dynamically adjust priorities and employs a cubic function for fine-grained differentiation. Additionally, it introduces a gain factor to accelerate the recovery of the transmission window. These measures collectively enable shorter flows to complete their transmission more rapidly.

Traditional TCP protocols tend to be conservative in their congestion control mechanisms and are slow to introduce and deploy new features, thereby facing certain limitations. To address this challenge, DCQUIC (Tan et al., 2021) has been proposed. DCQUIC represents a novel data center transport solution based on the QUIC protocol. It combines the efficiency of UDP with the reliability of TCP. DCQUIC supports modular design and connection migration techniques, offering the potential to significantly enhance the transmission performance and reliability of DCN.

Moreover, to address the Incast problem, which can lead to rapid buffer saturation in switches, resulting in packet loss and localized congestion, severely impacting network performance, a solution called AP (Zou et al., 2021b) has been proposed. AP dynamically adjusts the transmission intervals of TCP packets based on the number of concurrent flows to effectively mitigate TCP congestion, thereby reducing the likelihood of Incast issues occurring. On the Alibaba Cloud platform, the AP protocol was deployed on Elastic Compute Service (ECS) equipped with a 2.5G Intel Xeon CPU, 2 GB RAM, 40 GB hard disk, and a 100Mbps network interface card, running the CentOS 6.5 operating system. Experimental results show that, in environments with high concurrency or mild network congestion, enabling the AP protocol can significantly reduce the risk of timeouts and decrease bandwidth waste.

Current flow scheduling schemes typically prioritize flows based solely on their flow volume without fully considering the network's congestion status and available bandwidth. Additionally, existing congestion control schemes do not account for flow priorities, they apply the same congestion window adjustment strategy to both short and long flows, which can adversely affect the performance of short flows. To address these issues, the Flash (Gao et al., 2023) has been proposed. Flash takes into consideration the network's congestion status to adjust flow priorities and employs a priority-based congestion control strategy. At the switch side, Flash establishes multiple priority queues and enforces strict priority scheduling based on flow priorities. At the sender side, the congestion window size is adjusted based on network congestion signals and the number of bytes already sent by the flow, prioritizing the transmission of high-priority flows. Detailed information about the Flash workflow is presented in Fig. 11.

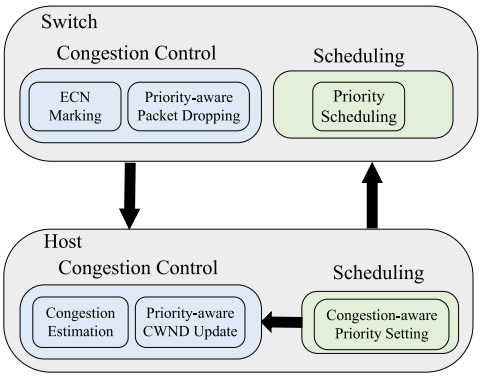


Fig. 11. Flash overview.

4.2.2. Active congestion control

Active congestion control algorithms primarily aim to prevent congestion in the network through proactive intervention and notifications to data senders. Unlike passive congestion control, active congestion control requires the participation of data senders and involves actively reducing their transmission rates or taking other measures based on indications of network congestion to reduce the likelihood of congestion occurring. The active congestion control algorithm is described in Table 5. SDN's centralized controller can collect and analyze real-time and historical information from all network switches and flows. However, the amount of data collected by the controller is substantial and requires fast computational capabilities for decision-making. This data can be used to analyze whether network paths are prone to congestion and configure optimal parameters for each TCP flow to prevent congestion. In this context, a congestion avoidance tool called RSCAT (Diel et al., 2022) is proposed. RSCAT utilizes reinforcement learning algorithms to dynamically detect and configure TCP transmission parameters in real-time, thereby automatically implementing congestion avoidance control for DCN flow.

The D^3 protocol (Wilson et al., 2011) is designed to meet the soft real-time requirements of user-interactive applications in data center networks by dynamically scheduling bandwidth and utilizing flow deadline information to ensure the timeliness of application responses. It addresses the insensitivity of traditional protocols like TCP to flow deadlines, adapting to the challenges posed by the concurrency of short flows and complex loads in data centers.

Traditional end-to-end congestion control based on TCP has become inadequate to meet the demands for low latency and high throughput within data centers. Therefore, there has been a gradual shift towards using Remote Direct Memory Access (RDMA). However, RDMA usage requires ensuring that packets are not lost to trigger Priority Flow Control (PFC). Nonetheless, PFC itself introduces a set of problems, such as head-of-line blocking. Hence, a switch-based congestion control scheme called RoCC (Menikkumbura et al., 2023) is proposed. RoCC's design incorporates an active computation of the fairness rate by switches, referred to as congestion points. These congestion points calculate fairness rate values in real-time using a Proportional-Integral (PI) control loop designed on the switch. Subsequently, it directly sends fairness rates to source nodes that need rate limiting in the form of ICMP packets, forming a closed-loop control system from source to congestion point and back to source.

While next-generation switch technologies offer higher bandwidth, they often come with relatively small buffers. This poses challenges for congestion control schemes relying on switch signals, such as ECN, which continuously adjust thresholds based on switch feedback. Therefore, a latency-control scheme for congestion control called Swift (Kumar et al., 2020) is introduced, which operates based on the principle of actively adjusting the congestion window. Swift measures end-to-end latency in real-time and compares it with a predefined target latency, proactively adjusting the sending rate to respond to network and endpoint congestion. The Swift protocol has significantly improved issues of large-scale concurrency and network congestion in Google's

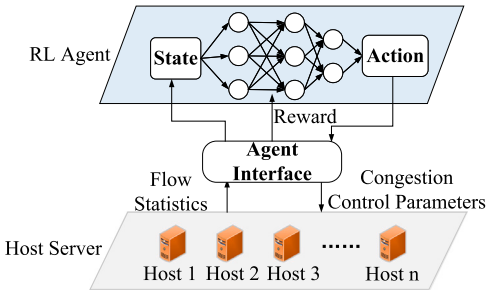


Fig. 12. PPO overview.

production environments, especially in handling high-density flow scenarios, such as large-scale incast situations. By flexibly adjusting the congestion window in response to network delays, Swift can effectively manage congestion in data center networks.

FACC (Chen et al., 2023) introduces a system that calculates congestion window thresholds based on the size of flow. It dynamically allocates congestion window sizes and prioritizes the processing of short flows based on flow size to meet the diverse requirements of different flow types in mixed workloads. It identifies short flows, such as small messages generated by parallel computing or machine learning tasks that need to be completed quickly, and long flows, such as large file transfers for Web services that require stable transmission. This approach optimizes the performance and efficiency of data center networks, ensuring that both short flows, which seek low flow completion times, and long flows, which require high throughput, receive services that meet their specific needs.

Different networks may require different optimal protocols and parameter settings, and conditions within a single network can change at any time. In the past, directly applying reinforcement learning to control congestion was infeasible because the learning cycle often exceeded the timescales of the network. PPO (Ketabi et al., 2023) proposes a framework based on multi-agent reinforcement learning that can dynamically adjust the parameters of traditional congestion control protocols. Through this approach, protocols can better adapt to changes in network conditions. Detailed information about the PPO workflow is presented in Fig. 12.

4.3. Discussion and insights on congestion control

In this section, we discuss passive congestion control and active congestion control, providing our insights. Firstly, passive congestion control algorithms are a category of algorithms that do not require active notification from the sender to adjust their sending rates. Instead, they adaptively control congestion by monitoring the network's state. Overall, these passive congestion control algorithms monitor network status and employ adaptive measures to achieve congestion control and performance optimization in DCN to some extent. However, these algorithms have limitations such as complexity, strong dependency on specific scenarios, and deployment challenges.

Secondly, active congestion control algorithms prevent congestion by actively notifying and intervening with the sender. Unlike passive congestion control algorithms that monitor the network and adapt, active congestion control algorithms require sender participation and cooperation to be effective. Active congestion control algorithms have their advantages over passive algorithms in certain aspects, but they also come with higher complexity. In conclusion, active congestion control algorithms achieve congestion prevention through active sender participation, providing clear effectiveness but adding complexity. They have their own characteristics compared to passive control algorithms, and ongoing research is needed to leverage the strengths of both and address their shortcomings.

We conducted quantitative experiments to discuss congestion control, utilizing the NS3 network simulation tool for simulations. This experiment discussed three network flow management schemes: DCTCP (Alizadeh et al., 2010), Swift (Kumar et al., 2020), and A^3DCT (Lu et al., 2023a), observing their impact on FCT under different network topologies and flow patterns. Specifically, the A^3DCT scheme demonstrated improvements in handling short-flow FCT across various network environments. In simulated bursty short-flow scenarios, A^3DCT achieved a reduction in the average FCT for short flows relative to DCTCP and Swift, reducing the average FCT for short flows by up to 16% and 11% compared to DCTCP and Swift, respectively. This result indicates that A^3DCT offers certain improvements in short-flow processing in specific scenarios. In large-scale real network topology scenarios with an 8-pod fat-tree, using a web search workload (Alizadeh et al., 2010), A^3DCT also showed performance improvements in handling short-flow workloads, achieving a 12.7% reduction in the average FCT for short flows compared to Swift. Furthermore, in mixed long-short flow network environments, A^3DCT's performance in terms of average FCT for short flows was superior to both DCTCP and Swift, improving by 60% and 42%, respectively (Lu et al., 2023a).

5. Routing optimization

In DCN, communication between endpoints holds a crucial position, necessitating the implementation of effective routing optimization strategies. In this section, we start by providing a clear definition of routing optimization, elucidating the core objectives of routing optimization. Subsequently, we provide a detailed overview of routing optimization solutions that have been developed in recent years in this field. Finally, we delve into a comprehensive discussion of these approaches.

5.1. Routing optimization overview

Routing optimization, as a critical component of DCN, plays a pivotal role. Its objective is to maximize the speed, availability, and reliability of data transmission while minimizing network congestion and latency (Li et al., 2021a). In the context of network modeling, we can represent the network as a directed graph, denoted as $G = (V, E)$, where V represents the network devices, such as hosts or switches, and E represents the links. The network's flow can be modeled as a flow matrix, where each element represents the flow demand between pairs of hosts. The routing matrix, denoted as R, is a representation of the network's routing algorithm. Each element of this matrix represents the proportion of flow distribution from flow requests to the network links. The product of flow requests and the routing matrix results in network link loads L, which is represented as a column vector with $|E|$ elements. R is a $|Q|\times|E|$ matrix, where $0 \le r_{qe} \le 1$, $q \in \{(i,j)|i,j \in N\}, e \in E$. The equation $L = Q \times R$ indicates that network load is primarily composed of two parts: the flow matrix and the routing matrix. Optimizing routing involves adjusting the routing matrix, which is the focus of routing optimization efforts. The matrix $R_{|Q|\times|E|}$ can be defined as:

$$R_{|Q|\times|E|} = \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,|E|} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,|E|} \\ \cdots & \cdots & \cdots & \cdots \\ r_{|Q|,1} & r_{|Q|,2} & \cdots & r_{|Q|,|E|} \end{bmatrix}. \tag{7}$$

Routing optimization aims to compute efficient paths for each host pair, facilitating the routing of corresponding flows. After path selection, the load l_{ij} on each link (i,j) is the sum of all flow requests passing through that link. The utilization of the link (i,j) is calculated as $U_{ij} = \frac{l_{ij}}{c_{ij}}$, where c_{ij} represents the capacity of link (i,j), and the available bandwidth on the link is given by $B_{ij} = c_{ij} - l_{ij}$. Routing optimization

can be formulated as the following equation, where the objective is to minimize routing costs when routing the load:

$$\min \phi = \sum_{(i,j)\in E} \phi_{ij}. \tag{8}$$

Here, ϕ_{ij} represents the routing cost on the path (i, j). Routing optimization aims to find the routing matrix R that minimizes the overall routing cost, taking into account the link loads U and available capacities B to ensure efficient and balanced network operation (Lee and Mukherjee, 2004).

The primary objectives of routing optimization are to ensure that data is transmitted from source to destination nodes in the most optimal manner while avoiding network congestion, reducing latency, and maximizing the utilization of network resources. We summarize several key objectives that routing optimization algorithms need to consider: **(1) The Impact of Routing Optimization Algorithms on FCT:** How do different routing optimization algorithms affect the FCT of data flows? Especially under various network load conditions, how do these algorithms ensure the rapid transmission of both small and large flows? **(2) The Relationship Between Throughput and Network Load Conditions:** How do routing optimization algorithms maximize the throughput of data flows under changing network load conditions? **(3) Quantitative Analysis of Link Load Balancing:** Utilizing the Gini coefficient to assess the balance of link loads, which routing optimization strategies can effectively mitigate the issue of unequal link loads? How can adjustments in algorithms lead to a more optimal load distribution? The introduction of the Gini coefficient to analyze the load distribution among different links is a valuable approach. The $Load\_G$ metric measures the imbalance in link loads, with higher values indicating greater load imbalance. The formula for $Load\_G$ is:

$$Load\_G = \frac{\sum_{i=1}^{L}\sum_{j=1}^{L}\left|\vartheta_i - \vartheta_j\right|}{2L^2\bar{\vartheta}}. \tag{9}$$

Where L is the total number of links, ϑ_i represents the load on link i, $\bar{\vartheta}$ is the average load across all L links. By calculating the Gini coefficient, you can quantify the degree of load imbalance among the links, helping to assess the level of load balancing achieved by the routing optimization scheme (Liu et al., 2021). A higher Gini coefficient suggests that some links carry a heavier load while others carry a lighter load, indicating the need for measures to improve load balancing.

5.2. Routing optimization schemes

Typically, routing optimization research primarily focuses on the internal aspects of DCN, as this is the core of data center operations, and optimizing internal routing can enhance the overall performance of the data center. However, external routing optimization is equally crucial for DCN. This involves routing flow between the data center and external networks, the internet, and other data centers. In Table 6, we provide a comprehensive overview of routing optimization solutions developed in recent years, covering both optimized routing in inter and intra DCN.

5.2.1. Optimize routing in inter DCN

Routing optimization within DCN primarily focuses on the optimization of network flow within data centers. This typically involves two main approaches, namely, the deadline-unaware routing scheme and the deadline-aware routing scheme. The deadline-unaware routing scheme concentrates on the optimization of network flow without the need to consider packet deadlines. On the other hand, the deadline-aware routing scheme requires an understanding of packet deadlines, as certain applications have strict timing requirements for packet delivery. Next, we will separately introduce the deadline-unaware routing scheme and the deadline-aware routing scheme.

FlowBender (Kabbani et al., 2014) employs an end-to-end flow-level load balancing mechanism that effectively reduces the average and tail latency in data center networks without the need for modifying hardware or relying on complex host mechanisms. It overcomes the limitations of traditional ECMP methods, such as imbalanced load distribution among long flows and path congestion, by utilizing end-to-end signals like ECN and timeout detection to identify congestion and failures, thereby achieving dynamic rerouting of flows.

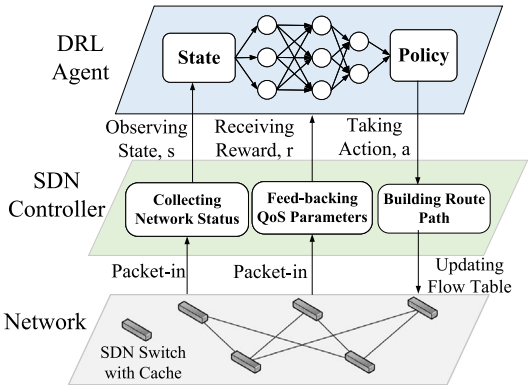


Fig. 13. DRL-R overview.

Traditional methods struggle to allocate network resources effectively to meet the performance requirements of different types of flow. To address this issue, the DRL-R (Liu et al., 2021) solution quantifies the contributions of caching and bandwidth to latency and combines them into schedulable and allocatable network resources. Subsequently, deep reinforcement learning is utilized for adaptive and intelligent routing decisions to optimize caching and bandwidth allocation for flow, thereby enhancing the routing performance of DCN. In Fig. 13, we provide a detailed overview of the DRL-R solution architecture.

The DRL-Plink (Liu et al., 2022b) solution aims to efficiently allocate DCN resources to meet the performance requirements of different types of flow. It achieves this by establishing private links to isolate different types of flow and applying deep reinforcement learning to adaptively and intelligently allocate bandwidth resources.

Some existing flow scheduling schemes, such as PIAS (Bai et al., 2015) and pFabric (Alizadeh et al., 2013) utilize Shortest Remaining Time First (SRTF) scheduling to minimize flow completion time. However, these schemes do not take into account the available network bandwidth, which can lead to suboptimal performance in dynamic DCN. To address this issue, the D-SRTF (Gao et al., 2021) scheme proposes a lightweight distributed approach for DCN. In this scheme, the remaining size of flows is estimated, and the available bandwidth is estimated using congestion indicators from congestion control to calculate the remaining time for each flow. Then, at the sender, packet priorities are set based on the remaining time and predefined thresholds to reduce the average flow completion time.

Existing deep reinforcement learning-based routing schemes face two challenges in large-scale networks: convergence difficulties and poor robustness. To address this issue, the ScaleDeep (Sun et al., 2022) scheme introduces the concept of selecting critical nodes from the network as driver nodes. DRL agents only need to observe the state of these driver nodes to approximate the entire network state. Subsequently, DRL agents can dynamically adjust the link weights of the driver nodes and modify routing strategies to enhance network performance.

The ORSM (Tang et al., 2022) scheme is focused on addressing the challenges posed by two different types of flow in DCN: deadline-aware flows and deadline-unaware flows. Existing mechanisms typically optimize performance metrics for only one of these flow types, making it difficult to simultaneously balance the performance of both. The ORSM scheme achieves dynamic determination of transmission paths and sending rates for each flow in mixed-flow scenarios within DCN

Table 6
Routing optimization schemes.

Ref.	Year	Evaluation method	Advantages	Disadvantages
FlowBender (Kabbani et al., 2014)	2014	NS3 for topology simulation, simulated flow testing	The design is simple	May result in packet reordering
DRL-R (Liu et al., 2021)	2021	OMNet++ for topology simulation, simulated flow testing	Supporting resource-aware joint scheduling	Lack of real network validation results
DRL-Plink (Liu et al., 2022b)	2022	Ryu as controller, Mininet for topology simulation, real network workload	Isolation of different types of flows using private link mechanisms	Performance in actual large-scale DCN needs further validation
D-SRTF (Gao et al., 2021)	2021	NS2 for topology simulation, real network workload	Simple and effective method for estimating flow information	Some impact on the performance of long flows
ScaleDeep (Sun et al., 2022)	2022	OMNet++ for topology simulation, simulated flow testing	Addressing the challenges of deep reinforcement learning in large-scale networks	Drive node selection algorithm lacks theoretical basis
ORSM (Tang et al., 2022)	2022	Custom simulation tool, real network workload	Real-time determination of paths and transmission rates using DCN feedback mechanisms	Limiting study in more complex topologies and flow scenarios
FHDM (Ampririt et al., 2023)	2023	Custom simulation tool	It can effectively handle multiple input parameters.	There are still differences from actual situations.
Salhab et al. (2020)	2020	Custom simulation tool, real network workload	Considered multiple objectives.	High computational complexity.
TED (Guo et al., 2022)	2022	Physical network devices for topology simulation	Joint optimization of route and egress selection	Long model training times
BATE (Zhang et al., 2021)	2023	Custom simulation tool, simulated flow testing	Stable performance in different network topologies, demand matrices, and routing schemes	Requiring access to global information, which may be challenging to obtain accurately in large-scale networks

by establishing a mathematical model and solving the routing and scheduling problem in a distributed manner.

N. Salhab et al. proposed a framework (Salhab et al., 2020) designed to optimize the mapping of 5G acquisition nodes to virtual machine pools in 5G software-defined data centers, with the aim of minimizing cloud computing costs and processing capacity while maximizing the network load on low-load acquisition nodes. The functional separation design adopted by 5G networks allows network functions to be deployed in software-defined data centers through virtualization, but the strategy for mapping acquisition nodes to virtual machine pools is crucial for achieving multi-objective optimization of costs, power consumption, and network load. The framework formalizes the mapping problem of 5G acquisition nodes to virtual machine pools as an integer linear programming problem, providing a dynamic mapping mechanism to balance and optimize these conflicting objectives.

FHDM (Ampririt et al., 2023) introduces a system based on fuzzy logic aimed at handling handover decisions in 5G networks, particularly considering the complexity of network slicing parameters. With the advancement of 5G technology, network management is confronted with challenges brought about by the introduction of new technologies such as SDN and network slicing. These technologies increase the complexity of mobile network management needed to meet the diverse Quality of Service (QoS) requirements of different applications. In this context, handover decisions need to account for a variety of wireless access technologies and a broad range of QoS requirements, while also dealing with multiple related but inconsistent parameters, constituting an NP-hard problem. The innovation of FHDM lies in its introduction of network slicing parameters into the handover decision process through fuzzy logic, proposing an intelligent system to manage this complex decision-making process, and verifying its effectiveness through experiments.

5.2.2. Optimize routing in intra DCN

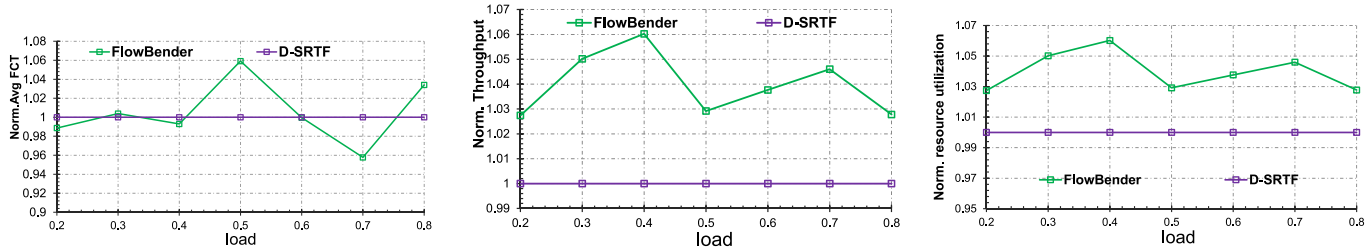
DCN external routing optimization involves routing flow that connects the data center to external networks, the internet, and other data centers. This encompasses network connections that span different geographical locations, service providers, and autonomous systems.

Online service providers are increasingly adopting shared Wide-Area Networks (inter-DC WANs) to reduce network infrastructure investments. These networks connect different data centers and Internet Service Providers (ISP). Such shared inter-DC WANs need to simultaneously carry two different priority levels of flow: ISP-facing flow and inter-DC flow. To address this challenge, the TED (Guo et al., 2022) approach is proposed. It is a mixed-integer nonlinear programming method that leverages deep reinforcement learning to train agents for the joint optimization of outbound route selection and path selection in the shared inter-DC WAN.

Wide Area Networks (WANs) connecting multiple geographically distributed data centers play a critical role, but they often face challenges due to limited bandwidth and frequent link failures, making it difficult to guarantee service availability. To address these issues, the BATE (Zhang et al., 2021) framework is proposed. BATE aims to provide bandwidth availability guarantees for cross-data-center WANs and consists of three core components: ingress control, flow scheduling, and fault recovery. It explicitly takes into account the failure probabilities of different links, and the fault recovery component handles actual failures efficiently. A testbed simulating a small inter-Data Center Wide Area Network (inter-DC WAN) among six data centers was constructed, operating at a rate of 1Gbps and a latency of 100 ms to mimic a WAN environment. Each server was equipped with four Intel Xeon E5-2620 CPUs, 64 GB of memory, and four Ethernet NICs, and ran 20 Virtual Machines (VM), all connected through OpenvSwitch. The VM operated on CentOS 7, using the Linux v4.15.6 kernel. Test results from the BATE framework showed that, across various topologies, flow patterns, and failure scenarios, it consistently outperformed existing flow engineering algorithms, with a Bandwidth Availability (BA) satisfaction rate under normal load conditions that was 23% to 60% higher than other algorithms.

5.3. Discussion and insights on routing optimization

Internal DCN flow optimization strategies can be broadly categorized into two main classes: deadline-unaware and deadline-aware approaches. In the context of deadline-unaware scenarios, existing



(a) Overall normalized average FCT (b) Normalized average throughput (c) Normalized average resource utilization

Fig. 14. Performance comparison of cache workload on routing optimization schemes.

research primarily focuses on optimizing network resource allocation to improve the performance of different types of flow. These methods significantly enhance network performance in scenarios involving mixed flow through resource optimization and intelligent scheduling. In deadline-aware scenarios, current research focuses on ensuring compliance with deadline requirements. Overall, current research still faces challenges such as the difficulty of validating methods' effectiveness in large-scale networks, insufficient robustness, and limited consideration of the impact of routing choices on scheduling outcomes.

In the realm of DCN, external routing optimization is a critical topic that involves optimizing flow routing between data centers and external networks, the Internet, and other data centers. This type of optimization task involves complex scenarios, including network connections spanning different geographic locations, service providers, and autonomous systems. As data centers continue to grow in scale and complexity, effectively managing and optimizing flow routing between data centers and external networks has become a significant challenge. By combining deep reinforcement learning, advanced algorithms, and a holistic approach to flow engineering, solutions like TED and BATE are contributing to the transformation of DCN into a more adaptive, efficient, and resilient ecosystem.

We conducted corresponding quantitative experiments to better analyze and discuss routing optimization. In this experiment, we utilize Ryu for control plane simulation and Mininet for data plane simulation, employing the cache workload (Roy et al., 2015) to simulate traffic. A 6-pod fat-tree topology was used as the network structure. This experiment explored two schemes: FlowBender (Kabbani et al., 2014) and D-SRTF (Gao et al., 2021). Similarly, here we normalize the test values of each scheme to the D-SRTF scheme. At lower levels of network load, FlowBender and D-SRTF demonstrated similar performance in terms of average FCT. However, under higher network load, D-SRTF exhibited lower FCT. Furthermore, D-SRTF improved throughput by 2.85% to 6.01% compared to FlowBender in Fig. 14(b). This performance gain benefits from the D-SRTF strategy's ability to predict the remaining size of a flow by estimating the number of bytes sent and utilizing congestion control indicators to estimate available bandwidth, thereby calculating each flow's remaining time, effectively increasing throughput. However, implementing this strategy requires more complex calculations to assess available bandwidth and calculate each flow's remaining time. This led to an average increase in CPU utilization for D-SRTF of 2% to 12% under higher network loads (0.5 to 0.8) compared to FlowBender in Fig. 14(c). This indicates that while D-SRTF shows advantages in enhancing network throughput and reducing FCT, it also demands more computational resources.

6. Flow scheduling

In the context of DCN, prioritizing efficient flow scheduling is essential for optimizing network performance. This section begins by defining the concept of flow scheduling. It then introduces the primary optimization objectives related to flow scheduling. Subsequently, an overview of recent flow optimization solutions and frameworks is provided. Finally, this section engages in discussion and offers insights on this topic.

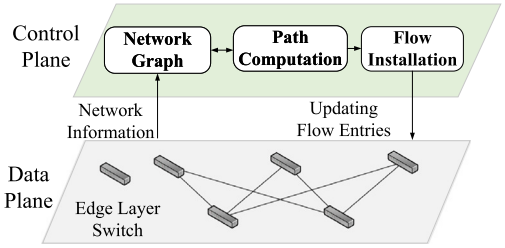


Fig. 15. Sieve overview.

6.1. Flow scheduling overview

DCN play a crucial role in modern information technology, serving as the foundational infrastructure connecting users, applications, and data storage. In this complex and highly interconnected network environment, the flow of data is as essential as blood flow within a living organism. In the following discussion, we will delve into how flow scheduling can address these challenges to build robust and dependable DCN. **(1) The Impact of Flow Scheduling Strategies on FCT:** In DCN, how do different flow scheduling strategies affect FCT of data flows? Particularly for flows of varying sizes and types, how do these strategies optimize FCT? **(2) Maximizing Network Throughput:** How do flow scheduling mechanisms maximize the throughput of a network while ensuring the fair use of network resources? **(3) Minimizing Maximum Link Utilization:** How can flow scheduling strategies be designed to minimize the utilization of the most congested link in a network?

6.2. Flow scheduling schemes

Next, we will begin by introducing centralized flow scheduling strategies and distributed flow scheduling strategies. In Table 7, we provide a detailed overview of recent flow scheduling schemes.

6.2.1. Centralized flow scheduling

Freeway (Wang et al., 2014) introduces a new architecture for flow management in data center networks, dynamically allocating paths between multiple shortest paths for low-latency and high-throughput, effectively distinguishing between large and small flows to resolve the competition issues inherent in traditional solutions. In data center networks with multi-rooted tree topologies, Freeway enhances overall network efficiency through real-time monitoring and path adjustments, achieving local load balancing for small flows and centralized scheduling for large flows. The Freeway scheme was evaluated in a large Fat-Tree topology simulating Facebook data center workloads, showcasing its ingenious design in effectively isolating large and small flows. This isolation strategy enables small flows to complete quickly, significantly reducing latency and thus improving user experience for applications with high real-time requirements, such as online meetings and video calls. At the same time, by allocating high-throughput paths

Table 7
Flow scheduling schemes.

Ref.	Year	Evaluation method	Advantages	Disadvantages
Freeway (Wang et al., 2014)	2016	Custom simulation tool with real workloads	Effectively isolating large flows and small flows	Incorrect identification can impact performance
Sieve (Zaher et al., 2021)	2021	Mininet simulation with real workloads	Dynamically scheduling flow based on port load is more flexible and efficient	Port load detection has some latency, may lead to optimization blind spots
Nougnanke et al. (2023)	2023	NS3 simulation with real workloads	Domain-agnostic, generalizes well to different topologies and parameter combinations	Slower prediction speed, real-time performance improvement needed
MiFi (Majidi et al., 2023)	2023	Mininet simulation with simulated flow	Considering global and local route reconfiguration	Performance in larger real networks requires further verification
ROAR (Guo et al., 2021)	2021	Real network topology and workloads	Introducing online intelligent routing using reinforcement learning	Learning algorithm parameters not extensively fine-tuned, may be stuck in local optima
Wang et al. (2022)	2022	Custom simulation tool with real workloads	The solution design is mature	The reward function design does not adequately consider the balance of all factors
DiffTREAT (Wei et al., 2023)	2023	NS3 simulation with real workloads	Utilizing deep learning for fine-grained differential scheduling	May require retraining models for new applications
Poche (Shen et al., 2022)	2022	NS3 simulation with simulated flow	Works well with caching for differentiated service objectives	Requiring additional cache resources on switches
P4TE (Robin and Khan, 2022)	2022	P4 and Mininet simulation with real workloads	Implementable on existing PISA switches	Flow rate control may increase TCP retransmissions
DRLIS (Wang et al., 2024)	2023	Real network environment and workloads	Optimization achieved load balancing, response time, and weighted cost goals.	Deep learning networks require ample training data, which could take time to gather.
DAMA (Xu et al., 2022)	2023	Custom simulation tool with simulation workloads	Applicable in real-world scenarios.	The communication model assumes equal bandwidth allocation, which may not reflect reality.

to large flows, their transmission becomes more efficient, ensuring that service quality is maintained even in situations of high network flow.

Traditional solutions struggle to simultaneously optimize the latency of small flows and the throughput of large flows. Therefore, Sieve (Zaher et al., 2021) proposes an SDN-based flow scheduling framework that samples a portion of network flow and dynamically reschedules large flows based on the bandwidth utilization of edge switch ports, as shown in the Fig. 15. This approach shortens the flow completion time for small flows while ensuring the throughput of large flows. The framework leverages the clustering feature of OpenFlow to partially sample network flow, thereby alleviating the controller's load.

Nougnanke et al. (2023) propose a machine learning-based performance prediction framework based on SDN. This framework leverages SDN's control plane global view and fine-grained telemetry capabilities to collect training data, including performance data and input feature data. When real-time flow metrics are input to the inference agent, it utilizes the trained machine learning models to make predictions and output estimated performance metric values. These prediction results can be used by the SDN control layer's intelligent caching management mechanisms or flow optimization algorithms to adjust system parameters in real-time to achieve efficient performance.

Within the context of SDN in DCN, the controller faces the critical task of periodically addressing the optimization problem related to Multi-Commodity Flow (MCF) and globally updating network configurations within extremely tight time constraints to maintain optimal flow engineering. However, new flow demands may arrive while the ongoing MCF optimization has not yet converged, making it difficult to respond promptly. To address these issues, the MiFi (Majidi et al., 2023) approach has been proposed. MiFi employs a propagation algorithm to dynamically determine the scope of reconfiguration and utilizes heuristics and update strategies based on Lyapunov control theory. These strategies serve to regulate the frequency of reconfiguration, achieving bounded updates to network flow within the data center while minimizing the expected flow cost under reconfiguration budget constraints.

Hybrid SDN refers to a network architecture where some SDN switches are deployed within a traditional network. However, current routing scheme designs often fail to adequately consider flow distribution or involve significant computational overhead in calculating routing schemes, making it challenging to respond swiftly to dynamically changing flow demands. To address this issue, ROAR (Guo et al., 2021) proposes an approach that utilizes reinforcement learning to train flow splitting agents. These agents establish a direct mapping between flow demands and flow splitting strategies, enabling rapid intelligent routing optimization for dynamic flow variations within hybrid SDN environments.

Y. Wang et al. proposed an energy efficiency management framework (Wang et al., 2022) using deep reinforcement learning, aimed at DCN to meet dynamic demands, achieving flow forecasting and optimization of the multi-commodity flow problem, facilitating dynamic topology configuration and minimizing energy consumption. The framework, through deep reinforcement learning, realizes real-time dynamic control of DCN, effectively improving energy efficiency and resource utilization.

6.2.2. *Distributed flow scheduling*

DiffTREAT (Wei et al., 2023) employs deep learning using Recurrent Neural Networks (RNN) (Sherstinsky, 2020) and network caching to optimize data transmission within data centers. This scheme leverages deep learning techniques to classify flow and predict flow sizes, obtaining the flow feature information required for differential scheduling. Simultaneously, a differential scheduling method based on a multi-level priority queue is introduced, considering both flow type and size as dimensions for priority allocation.

DCN host a variety of flow types, necessitating support for deep differentiation services to meet the demands of diverse applications. Hence, a priority-based and flow-aware in-network caching scheme called Poche (Shen et al., 2022) is proposed. This scheme achieves deep differentiation by introducing additional cache resources within switches.

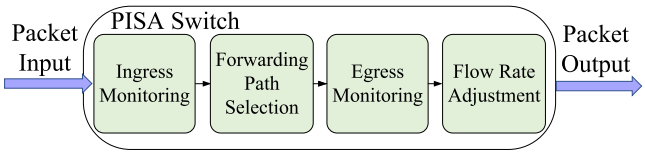


Fig. 16. P4TE workflow.

Recently, programmable switches based on the PISA architecture have emerged, boasting formidable computational capabilities. P4TE (Robin and Khan, 2022) represents a flow engineering system tailored for DCN based on fat-tree topologies, as shown in Fig. 16. It employs P4 programming to implement link monitoring, load-aware path selection, and flow rate control in the data plane. Importantly, it can be deployed on existing programmable switches, thereby enhancing network resource utilization and flow completion times.

DRLIS (Wang et al., 2024) is a scheduling algorithm based on deep reinforcement learning, designed for networked devices in edge and cloud computing environments. DRLIS addresses these challenges by integrating a load balancing model with an application response cost model, and employing the Proximal Policy Optimization (PPO) algorithm to expedite optimization. This approach effectively mitigates the limitations of edge computing devices and the latency issues associated with cloud computing.

X. Yuzhe et al. proposed an innovative Distributed Auction for Multiple Assignment (DAMA) (Xu et al., 2022) to address the allocation and load balancing of Deep Neural Network (DNN) inference tasks in edge networks. DAMA introduces a detailed DNN inference task allocation process model aimed at efficiently distributing tasks and balancing loads in edge environments. This method involves decomposing the DNN model into multiple modules and then distributing these modules across several edge servers for joint execution, significantly reducing inference time and enhancing efficiency.

6.3. Discussion and insights on flow scheduling

In recent years, the academic community has proposed various flow scheduling schemes to optimize performance metrics such as network flow completion time, throughput, and link utilization. These schemes are designed from two perspectives: centralized and distributed. Both centralized and distributed scheduling schemes have their advantages and drawbacks. Future flow scheduling solutions should aim to integrate the strengths of both approaches and build upon this foundation for further improvement.

Here, we carry out quantitative analysis to better discuss flow scheduling strategies. This experiment discussed two schemes, Freeway (Wang et al., 2014) and Sieve (Zaher et al., 2021), which propose different solutions to the competition between large and small flows in networks. We conducted network simulations using Ryu for the control plane and Mininet for the data plane, adopting a 6-pod fat-tree topology. The simulations employed a cache workload (Roy et al., 2015) to mimic flow patterns. Similarly, here we normalize the test values of each scheme to the Freeway scheme. Performance comparisons indicate that Freeway performs better in terms of average FCT than Sieve, with improvements ranging from a low of 0.41% to a high of 6.41% in Fig. 17(a), particularly in handling small flows (mouse flows) in Fig. 17(e), where it reduces the average by 3.33% to 9.04%, and in dealing with large flows (elephant flows), where it reduces the average by 7.23% to 15.17% in Fig. 17(f). Moreover, Freeway also exhibits an improvement in average throughput, ranging from 0.22% to 2.31% in Fig. 17(b). These achievements are attributed to Freeway's effective use of its dynamic path allocation mechanism, distinguishing between low-latency paths and high-throughput paths to accommodate different flow demands, ensuring efficient flow transmission. In terms of CPU utilization, under high network load levels (0.7 to 0.8), Sieve's CPU utilization is on average 2.04% to 8.8% lower than that of Freeway in Fig. 17(c). This difference stems from Freeway's need for real-time dynamic calculations to adjust paths and monitor path latency, whereas Sieve reduces the controller's load by sampling a portion of network flow, thereby lowering overall resource consumption.

7. Flow security

In DCN, the issue of flow security is increasingly prominent in its importance. This section firstly outlines the fundamental concepts of flow security, followed by a detailed introduction to the primary optimization objectives of flow security. Subsequently, solutions related to flow security are explored. Finally, we present our own specific insights into these solutions.

7.1. Flow security overview

Flow security is crucial in data center management, as it directly relates to whether the data center can effectively resist network attacks and ensure the integrity, availability, and confidentiality of data. To maintain the security and stability of the network environment, the following key objectives for flow security need to be considered: **(1) Defense Strategies Against External Attacks:** In DCN, which defense strategies are most effective against various types of external attacks, such as DDoS attacks and intrusion attempts? Considering the diversity and complexity of attack methods, how can these strategies be integrated to enhance defense efficiency? **(2) Enhancing Network Resilience:** What mechanisms ensure that a network can rapidly recover to its normal operational state in the event of an attack? **(3) Minimizing Performance Impact:** How to balance security and network performance when implementing flow security strategies? Specifically, how to ensure the integrity, availability, and confidentiality of data without significantly affecting network throughput and response times?

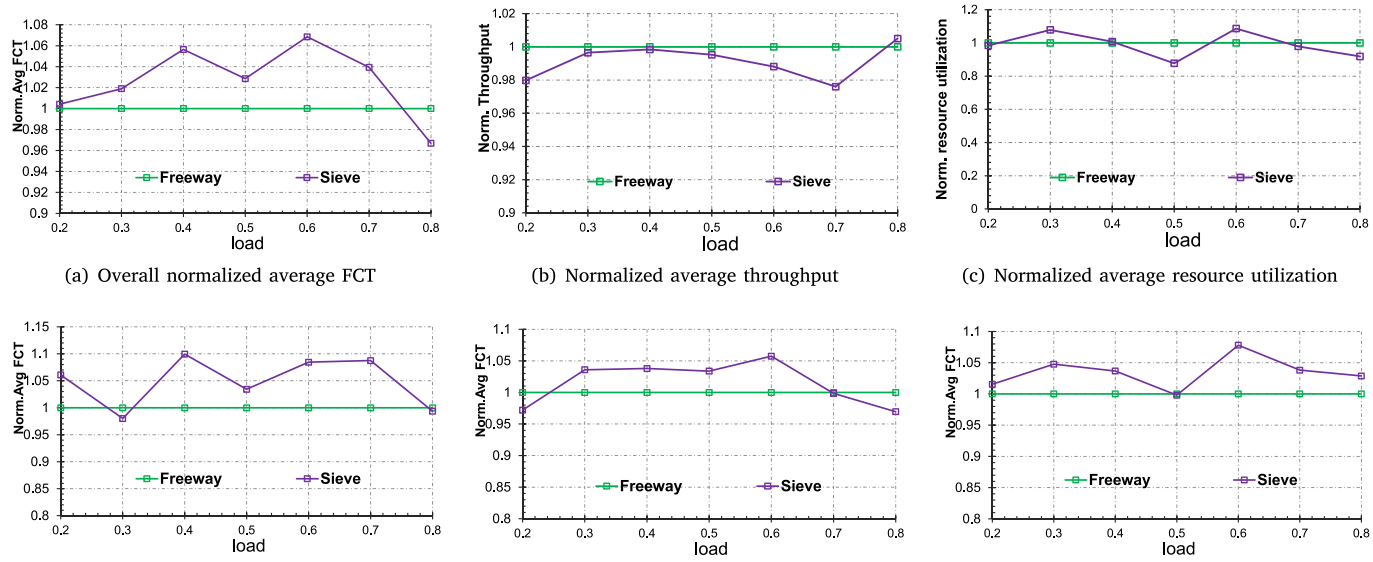
7.2. Flow security schemes

Next, we will introduce flow security strategies related to intrusion detection and DDoS prevention. In Table 8, we provide a detailed description of the latest developments in flow security.

7.2.1. Intrusion detection

SDN greatly simplifies network management processes through the decoupling of the control plane and the data plane. However, SDN introduces a series of new security challenges, among which Distributed Denial of Service (DDoS) attacks are particularly prominent, posing a serious threat to the stability and reliability of SDN networks. The two-stage DDoS attack detection system (Niu et al., 2023) developed by M. Niu et al. is specifically designed for SDN networks, achieving efficient and accurate attack detection by combining multi-feature and adaptive threshold triggering mechanisms with a Support Vector Machine (SVM) classifier. In the first stage, the system dynamically adjusts thresholds to adapt to network changes, significantly enhancing flexibility and accuracy compared to traditional methods that rely on a single feature and static thresholds. The SVM classifier in the second stage further verifies and identifies DDoS attacks, ensuring the precision of detection. The application of this two-stage method in the SDN environment enhances detection efficiency and significantly reduces resource consumption.

The DDoS attack detection model (El Sayed et al., 2022) developed by M. Sayed et al. is specifically designed for the SDN environment, incorporating deep learning techniques and advanced feature selection methods. By utilizing Long Short-Term Memory (LSTM) networks and autoencoders, this model is capable of effectively identifying abnormal flow patterns in SDN. The model extracts 48 key statistical features from SDN flow data and uses information gain and random forest



(a) Overall normalized average FCT (b) Normalized average throughput (c) Normalized average resource utilization

(d) Mice flow normalized average FCT (0, 100 KB) (e) Medium flow normalized average FCT (100 KB, 10 MB) (f) Elephant flow normalized average FCT (10 MB, +∞)

Fig. 17. Performance comparison of cache workload on flow scheduling schemes.

Table 8
Flow security schemes.

Ref.	Year	Evaluation method	Advantages	Disadvantages
Niu et al. (2023)	2023	Mininet simulation with public datasets	Less consumption of controller resources	Even in extremely idle moments, the detection module will still be called multiple times.
El Sayed et al. (2022)	2022	Mininet simulation with public datasets	Deep learning methods can automatically learn features	Performing offline analysis solely on the flow
Zainudin et al. (2022)	2023	Mininet simulation with public datasets	A hybrid use of CNN and LSTM enables comprehensive learning of flow features	The robustness of the model during long-term operation remains to be observed
Priyadarsini et al. (2022)	2023	Mininet simulation with simulated flow	It can effectively detect various types of attacks	Partial unknown attacks may introduce bias, requiring continuous model improvements
Zhou et al. (2021)	2021	Mininet simulation with simulated flow	Effectively misleading attackers and altering the attack surface	Considering only external attacks
POSEIDON (Li et al., 2021b)	2021	P4 simulation with simulated flow	Highly efficient performance and simplified deployment	Relies on programmable switches

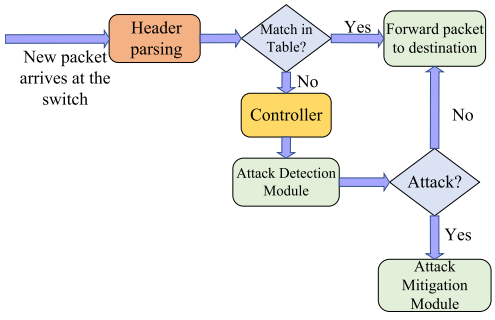


Fig. 18. An intrusion detection framework by M. Sayed et al..

methods to filter out the top 10 most influential features, thereby enhancing the accuracy and efficiency of detection. This approach, which combines deep learning with refined feature selection, significantly improves the model's generalizability and accuracy across different datasets. Detailed information about the framework of the intrusion detection framework by M. Sayed et al. in Fig. 18.

A. Zainudin et al. proposed an efficient detection and classification method for DDoS attacks, combining XGBoost feature selection and a CNN-LSTM hybrid deep learning model (Zainudin et al., 2022). This approach starts with feature selection via the XGBoost algorithm, effectively reducing feature dimensions and improving the model's efficiency and accuracy. Subsequently, a CNN-LSTM hybrid model is employed, where the two-dimensional Convolutional Neural Network (CNN) is responsible for extracting the spatial characteristics of features, while the LSTM network captures the dependencies of these features over time. This combination leverages the powerful capabilities of deep learning while taking into account the strict requirements for low latency and computational efficiency.

7.2.2. Ddos prevention

M. Priyadarsini et al. proposed a two-stage security enhancement strategy (Priyadarsini et al., 2022) for SDN controllers to address the security challenges. While the centralized nature of SDN simplifies network management, it also introduces new security risks, such as Denial of Service attacks and controller hijacking. Existing solutions are mostly focused on defending against Denial of Service attacks and lack detection for multiple types of attacks. To address this, the study introduced a Security Enforcement Framework (SEF), which combines

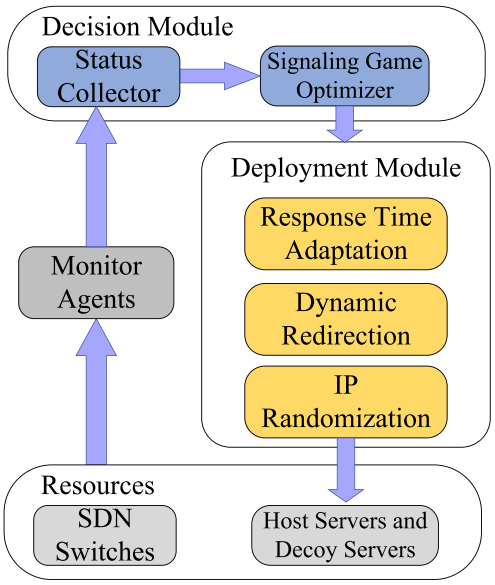


Fig. 19. An overview of the DDoS defense framework by Y. Zhou et al..

Trust-based Controller Attack Detection (TCAD) and Risk-based Attack Prevention (RAP) models. TCAD utilizes signaling game theory to analyze communication behavior for proactive attack detection, introducing time epochs to adjust trust levels. RAP assesses risks from the application and protocol layers to control malicious flow. This provides a comprehensive proactive defense for SDN.

Y. Zhou et al. proposed a lightweight hybrid DDoS defense framework (Zhou et al., 2021) based on SDN technology, integrating Moving Target Defense (MTD) and network deception techniques. Faced with the limitations of traditional DDoS defense methods such as blackhole routing and intrusion detection systems, this framework aims to enhance network security and attack resistance through a hybrid strategy. By utilizing an explicit signaling game model to optimize defense strategies, the framework confuses attackers, making it difficult for them to distinguish between real and fake servers, thereby strengthening defense. Detailed information about the overview of the DDoS defense framework by Y. Zhou et al. in Fig. 19.

POSEIDON is a high-performance, low-cost, and flexible system (Li et al., 2021b) aimed at defending against the increasingly severe large-scale DDoS attacks. POSEIDON employs next-generation programmable switches to introduce a system-level DDoS defense solution, balancing the advantages of new technologies with the simplification of hardware complexity. The system utilizes a high-level policy language to simplify the expression and deployment of DDoS defense strategies. Through optimized resource orchestration mechanisms, it efficiently maps strategies to hardware, optimizing performance to withstand strong attacks.

7.3. Discussion and insights on flow security

The development of SDN has introduced new security risks, especially DDoS attacks. To address these challenges, researchers have proposed efficient detection and prevention methods based on machine learning and deep learning, specifically designed for the SDN environment. These studies demonstrate that by integrating machine learning, optimization techniques, and proactive defense measures with the concept of moving targets, the security of the SDN environment can be significantly improved, effectively addressing the evolving security threats. In the future, as SDN technology continues to mature and evolve, advanced security strategies combining artificial intelligence and machine learning will become increasingly important. They will further enhance the network's self-defense capabilities, bringing revolutionary changes to the field of network security.

Here, we conduct quantitative analysis to facilitate a better discussion on flow security strategies. This experiment discusses the scheme by Zainudin et al. (2022), which compared the performance of a deep learning-based Hybrid-DNN model with other Machine Learning (ML) and Deep Learning (DL) methods in DDoS attack detection and classification. The proposed model leverages the strengths of CNN and LSTM, employing XGBoost feature selection technique for efficient DDoS attack detection and classification. Comparative schemes included two machine learning-based models (CART decision trees) and five deep learning-based models (CNN, Deep CNN, LSTM, GRU, and an existing CNN-LSTM). Under the same network configuration, performance was evaluated by comparing accuracy, computation time, the number of learning parameters, and memory computation (MFLOPs). The results demonstrated that the proposed Hybrid-DNN model achieved a 99.50% accuracy in DDoS attack detection and classification, surpassing all other models. Compared to the existing best model (an existing CNN-LSTM, with an accuracy of 99.44% and a time cost of 0.218 ms), the proposed model not only improved accuracy to 99.50% but also reduced the time cost to 0.179 ms, significantly lowered the number of learning parameters to 4.83K (from 32.81K in the existing CNN-LSTM), and decreased the computational complexity (MFLOPs reduced from 0.520 in the existing model to 0.067) (Zainudin et al., 2022).

8. Future development trends and prospects

In this section, we discuss the future development trends and prospects for flow optimization strategies, covering load balancing, congestion control, routing optimization, flow scheduling and flow security one by one.

8.1. Load balancing

Customized load balancing: Load balancing is becoming highly customizable, aiming to cater specifically to different applications by considering factors like the application's nature, user locations, and network conditions. Real-time apps focus on low latency and high bandwidth, whereas data processing apps prioritize throughput and resource maximization. The use of SDN and Network Functions Virtualization (NFV) introduces new enhancements in load balancing. SDN makes network management more flexible, and NFV allows for quick service rollouts and scalable growth, improving the network's responsiveness to changing demands. This tailored approach and technological integration will improve network management and optimization for diverse applications.

Multi-objective optimization: Future load balancing strategies will broaden their focus from just one performance metric to a holistic view, considering various metrics like efficiency, latency, energy usage, security, and more. Using advanced multi-objective optimization, these systems will make smarter flow allocation decisions, aiming to find the best balance among these different metrics.

Integration of machine learning and reinforcement learning algorithms: Load balancing strategies are increasingly adopting machine learning and reinforcement learning algorithms. These algorithms have the capability to autonomously make decisions and optimize load balancing in response to real-time network flow and application requirements. An example is using deep learning to recognize flow patterns, forecast future loads, and adjust strategies accordingly.

8.2. Congestion control

Real-time monitoring and prediction of network states: Congestion control is shifting focus to prioritize real-time monitoring and predicting network conditions. This means network devices and algorithms will actively monitor metrics such as flow, utilization, and latency, using this data to predict possible congestion. Through real-time observation and forecasting, networks aim to spot potential problems early and take steps to improve performance and availability without missing any content.

Proactive congestion avoidance: Congestion control is increasingly focusing on proactive avoidance, taking preventive actions to stop congestion before it happens rather than just reacting to it. This includes adjusting flows, changing routes, and optimizing resources based on real-time data and forecasts to keep the network stable.

AI-driven congestion control: As AI technology advances, AI-driven congestion control is emerging as a key trend. AI and machine learning offer more accurate network flow predictions, leading to more adaptable congestion control methods. These technologies enable networks to auto-adjust to varying flow patterns, improving network efficiency and stability.

8.3. Routing optimization

Coordinated optimization of computing, storage, and network resources: Routing optimization is evolving to consider computing, storage, and network resources together. Decisions will factor in not just network layout and traffic but also the distribution and needs of computing and storage resources. For example, choices could be based on server workloads and storage needs to pick the best path, enhancing overall data center efficiency and resource use.

Introduction of network intent awareness: Routing optimization is evolving to consider computing, storage, and network resources together. Decisions will factor in not just network layout and traffic but also the distribution and needs of computing and storage resources. For example, choices could be based on server workloads and storage needs to pick the best path, enhancing overall data center efficiency and resource use.

Applications of machine learning algorithms: Using machine learning in routing optimization helps networks more accurately predict conditions, leading to better routing decisions. These algorithms process large data sets to find patterns and trends, allowing for automatic adjustments to routing strategies to enhance network performance and resource use.

8.4. Flow scheduling

Improving algorithm generalization and adaptation to diverse network environments: Flow scheduling strategies are moving towards improving algorithm adaptability across different network settings, including data center networks, wide-area networks, and 5G networks, with their unique topologies and traits. This goal requires algorithms to better understand and adapt to network dynamics and variability, ensuring they work well in a variety of conditions.

Providing end-to-end scheduling guarantees to prevent performance anomalies: Flow scheduling is increasingly focusing on providing end-to-end scheduling guarantees. This means algorithms should aim not just to optimize individual nodes or links but to ensure the performance across the entire path or service chain. Such a comprehensive approach is key to preventing performance issues, properly managing network flows, and meeting specific QoS standards.

SDN/NFV integration: The deeper integration of SDN and NFV opens new possibilities for flow scheduling. SDN offers flexible network management and configuration, while NFV allows for quick service deployment and more adaptable network services. Together, SDN and NFV can lead to more effective flow scheduling, addressing changing network needs and business policies efficiently.

8.5. Flow security

Integrating advanced artificial intelligence and machine learning: With the progress in AI and machine learning, future flow security strategies are expected to grow more intelligent and automated. Advanced AI models like deep learning and reinforcement learning will be used to dynamically recognize and adjust to new security threats, improving the precision and efficiency of IDS and defense mechanisms. AI-driven security systems will learn on their own, continuously improving defense strategies against increasingly sophisticated attack patterns.

Adaptive security strategies and dynamic defense mechanisms: As cyber-attack methods keep evolving, future flow security solutions will have to become more adaptable and flexible. Security strategies and defenses that can auto-adjust using real-time network status and threat analysis will be crucial. Such strategies might cover risk-based security measures, adaptive access controls, and dynamic resource management, all aiming to maintain network stability and performance amidst diverse attack situations.

Cross-domain security collaboration: With the convergence of IoT, edge computing, and cloud computing, future networks will be more complex and widespread. Flow security strategies will have to cover various network layers and areas, enabling security collaboration across devices and platforms. This necessitates creating security protocols and standards suitable for distributed environments and mechanisms for sharing threat intelligence across domains.

9. Conclusion

This article explores flow optimization strategies in DCN during the digital era. DCN is essential for supporting cloud computing, big data analytics, and online services. With increasing data transmission and processing demands, DCN faces significant performance and efficiency challenges. Traditional optimization strategies often fall short due to DCN's unique flows and complex topologies. We review the latest research and developments in DCN flow optimization, covering load balancing, congestion control, routing optimization, flow scheduling, and security. We discuss the strengths and weaknesses of various approaches, showcasing the field's complexity and diversity. The article underscores the ongoing research efforts to address DCN's growing needs. As DCN evolves, so will flow optimization strategies, prompting continued exploration of new methods and technologies to meet DCN challenges.

CRediT authorship contribution statement

Yong Liu: Writing – original draft, Writing – review & editing. **Tianyi Yu:** Validation, Writing – original draft. **Qian Meng:** Resources, Supervision. **Quanze Liu:** Investigation, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

Abdelmoniem, A.M., Bensaou, B., 2021. T-RACKs: A faster recovery mechanism for TCP in data center networks. IEEE/ACM Trans. Netw. 29 (3), 1074–1087.

Al-Fares, M., Loukissas, A., Vahdat, A., 2008. A scalable, commodity data center network architecture. ACM SIGCOMM Comput. Commun. Rev. 38 (4), 63–74.

Alizadeh, M., Greenberg, A., Maltz, D.A., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., Sridharan, M., 2010. Data center tcp (dctcp). In: Proceedings of the ACM SIGCOMM 2010 Conference. pp. 63–74.

Alizadeh, M., Yang, S., Sharif, M., Katti, S., McKeown, N., Prabhakar, B., Shenker, S., 2013. pfabric: Minimal near-optimal datacenter transport. ACM SIGCOMM Comput. Commun. Rev. 43 (4), 435–446.

Ampririt, P., Higashi, S., Qafzezi, E., Ikeda, M., Matsuo, K., Barolli, L., 2023. An intelligent fuzzy-based system for handover decision in 5G-IoT networks considering network slicing and SDN technologies. Internet Things 23, 100870.

Bai, W., Chen, L., Chen, K., Han, D., Tian, C., Wang, H., 2015. Information-agnostic flow scheduling for commodity data centers. In: 12th USENIX Symposium on Networked Systems Design and Implementation. NSDI 15, pp. 455–468.

Bai, W., Chen, L., Chen, K., Wu, H., 2016. Enabling ECN in multi-service multi-queue data centers. In: 13th USENIX Symposium on Networked Systems Design and Implementation. NSDI 16, Santa Clara, CA, pp. 537–549.

Bai, W., Hu, S., Chen, K., Tan, K., Xiong, Y., 2021. One more config is enough: Saving (DC)tcp for high-speed extremely shallow-buffered datacenters. IEEE/ACM Trans. Netw. 29 (2), 489–502.

Benson, T., Akella, A., Maltz, D.A., 2010a. Network traffic characteristics of data centers in the wild. In: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement. pp. 267–280.

Benson, T., Anand, A., Akella, A., Zhang, M., 2010b. Understanding data center traffic characteristics. ACM SIGCOMM Comput. Commun. Rev. 40 (1), 92–99.

Bilal, K., Khan, S.U., Kolodziej, J., Zhang, L., Hayat, K., Madani, S.A., Min-Allah, N., Wang, L., Chen, D., 2012. A comparative study of data center network architectures. In: ECMS. pp. 526–532.

Chen, G., Han, J., Jie, X., Hong, P., Xue, K., 2023. FACC: Flow-size-aware congestion control in data center networks. In: 2023 IEEE Symposium on Computers and Communications. ISCC, IEEE, pp. 348–353.

Cheng, Y., Jia, X., 2020. NAMP: Network-aware multipathing in software-defined data center networks. IEEE/ACM Trans. Netw. 28 (2), 846–859.

Chung, W.-K., Li, Y., Ke, C.-H., Hsieh, S.-Y., Zomaya, A.Y., Buyya, R., 2023. Dynamic parallel flow algorithms with centralized scheduling for load balancing in cloud data center networks. IEEE Trans. Cloud Comput. 11 (1), 1050–1064.

D'Ambrosia, J., Law, D., Nowell, M., 2008. 40 Gigabit ethernet and 100 gigabit ethernet technology overview.

Dholakiya, D., Kshirsagar, T., Nayak, A., 2021. Survey of mininet challenges, opportunities, and application in software-defined network (sdn). In: Information and Communication Technology for Intelligent Systems: Proceedings of ICTIS 2020, Volume 2. Springer, pp. 213–221.

Di, S., Kondo, D., Cappello, F., 2013. Characterizing cloud applications on a Google data center. In: 2013 42nd International Conference on Parallel Processing. IEEE, pp. 468–473.

Diel, G., Miers, C.C., Pillon, M.A., Koslovski, G.P., 2022. Data classification and reinforcement learning to avoid congestion on SDN-based data centers. In: GLOBECOM 2022 - 2022 IEEE Global Communications Conference. pp. 2861–2866.

Dixit, A., Prakash, P., Hu, Y.C., Kompella, R.R., 2013. On the impact of packet spraying in data center networks. In: 2013 Proceedings IEEE INFOCOM. pp. 2130–2138.

El Sayed, M.S., Le-Khac, N.-A., Azer, M.A., Jurcut, A.D., 2022. A flow-based anomaly detection approach with feature selection method against ddos attacks in sdns. IEEE Trans. Cogn. Commun. Netw. 8 (4), 1862–1880.

Gao, C., Chu, S., Xu, H., Xu, M., Ye, K., Xu, C., 2023. Flash: Joint flow scheduling and congestion control in data center networks. IEEE Trans. Cloud Comput. 11 (1), 1038–1049.

Gao, C., Lee, V.C., Li, K., 2021. D-SRTF: Distributed shortest remaining time first scheduling for data center networks. IEEE Trans. Cloud Comput. 9 (2), 562–575.

Greenberg, A., Hamilton, J.R., Jain, N., Kandula, S., Kim, C., Lahiri, P., Maltz, D.A., Patel, P., Sengupta, S., 2009. VL2: A scalable and flexible data center network. In: Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication. pp. 51–62.

Guo, C., Lu, G., Li, D., Wu, H., Zhang, X., Shi, Y., Tian, C., Zhang, Y., Lu, S., 2009. Bcube: a high performance, server-centric network architecture for modular data centers. In: Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication. pp. 63–74.

Guo, Y., Ma, Y., Luo, H., Wu, J., 2022. Traffic engineering in a shared inter-DC WAN via deep reinforcement learning. IEEE Trans. Netw. Sci. Eng. 9 (4), 2870–2881.

Guo, Y., Wang, W., Zhang, H., Guo, W., Wang, Z., Tian, Y., Yin, X., Wu, J., 2021. Traffic engineering in hybrid software defined network via reinforcement learning. J. Netw. Comput. Appl. 189, 103116.

Hammadi, A., Mhamdi, L., 2014. A survey on architectures and energy efficiency in Data Center Networks. Comput. Commun. 40, 1–21.

Hu, J., Huang, J., Lyu, W., Li, W., Li, Z., Jiang, W., Wang, J., He, T., 2021. Adjusting switching granularity of load balancing for heterogeneous datacenter traffic. IEEE/ACM Trans. Netw. 29 (5), 2367–2384.

Huang, S., Dong, D., Bai, W., 2018. Congestion control in high-speed lossless data center networks: A survey. Future Gener. Comput. Syst. 89, 360–374.

Huang, J., Lyu, W., Li, W., Wang, J., He, T., 2021. Mitigating packet reordering for random packet spraying in data center networks. IEEE/ACM Trans. Netw. 29 (3), 1183–1196.

Kabbani, A., Vamanan, B., Hasan, J., Duchene, F., 2014. 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. pp. 149–160.

Kandula, S., Katabi, D., Sinha, S., Berger, A., 2007. Dynamic load balancing without packet reordering. SIGCOMM Comput. Commun. Rev. 37 (2), 51–62.

Kang, N., Ghobadi, M., Reumann, J., Shraer, A., Rexford, J., 2015. Efficient traffic splitting on commodity switches. In: Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies. pp. 1–13.

Ketabi, S., Chen, H., Dong, H., Ganjali, Y., 2023. A deep reinforcement learning framework for optimizing congestion control in data centers. In: NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium. pp. 1–7.

Kishor, A., Niyogi, R., Chronopoulos, A.T., Zomaya, A.Y., 2023. Latency and energy-aware load balancing in cloud data centers: A bargaining game based approach. IEEE Trans. Cloud Comput. 11 (1), 927–941.

Kreutz, D., Ramos, F.M., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S., Uhlig, S., 2014. Software-defined networking: A comprehensive survey. Proc. IEEE 103 (1), 14–76.

Kumar, G., Dukkipati, N., Jang, K., Wassel, H.M., Wu, X., Montazeri, B., Wang, Y., Springborn, K., Alfeld, C., Ryan, M., et al., 2020. Swift: Delay is simple and effective for congestion control in the datacenter. 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. pp. 514–528.

Lee, Y., Mukherjee, B., 2004. Traffic engineering in next-generation optical networks. IEEE Commun. Surv. Tutor. 6 (3), 16–33.

Li, Y., Guo, X., Pang, X., Peng, B., Li, X., Zhang, P., 2020. Performance analysis of floodlight and ryu SDN controllers under mininet simulator. In: 2020 IEEE/CIC International Conference on Communications in China. ICCC Workshops, pp. 85–90.

Li, W., Liu, J., Wang, S., Zhang, T., Zou, S., Hu, J., Jiang, W., Huang, J., 2021a. Survey on traffic management in data center network: from link layer to application layer. IEEE Access 9, 38427–38456.

Li, G., Zhang, M., Wang, S., Liu, C., Xu, M., Chen, A., Hu, H., Gu, G., Li, Q., Wu, J., 2021b. Enabling performant, flexible and cost-efficient DDoS defense with programmable switches. IEEE/ACM Trans. Netw. 29 (4), 1509–1526.

Liu, W.-x., Cai, J., Chen, Q.C., Wang, Y., 2021. DRL-R: Deep reinforcement learning approach for intelligent routing in software-defined data-center networks. J. Netw. Comput. Appl. 177, 102865.

Liu, W.-X., Cai, J., Zhu, Y.-H., Luo, J.-M., Li, J., 2023a. Load balancing inside programmable data planes based on network modeling prediction using a GNN with network behaviors. Comput. Netw. 227, 109695.

Liu, J., Huang, J., Li, W., Wang, J., He, T., 2023b. Asymmetry-aware load balancing with adaptive switching granularity in data center. IEEE/ACM Trans. Netw. 31 (3), 1145–1158.

Liu, J., Huang, J., Lv, W., Wang, J., 2022a. APS: Adaptive packet spraying to isolate mix-flows in data center network. IEEE Trans. Cloud Comput. 10 (2), 1038–1051.

Liu, W.-X., Lu, J., Cai, J., Zhu, Y., Ling, S., Chen, Q., 2022b. DRL-PLink: Deep reinforcement learning with private link approach for mix-flow scheduling in software-defined data-center networks. IEEE Trans. Netw. Serv. Manag. 19 (2), 1049–1064.

Liu, S., Xu, H., Cai, Z., 2013. Low latency datacenter networking: A short survey. arXiv preprint arXiv:1312.3455.

Liu, K., Zhang, J., Wei, D., Zhang, K., Huang, T., 2020. PLB: Adaptive partial congestion-aware load balancing for datacenter networks. In: GLOBECOM 2020 - 2020 IEEE Global Communications Conference. pp. 1–6.

Lu, Y., Cui, C., Ma, X., Ruan, Z., 2023a. A3DCT: A cubic acceleration TCP for data center networks. J. Netw. Comput. Appl. 216, 103654.

Lu, Y., Xu, Z., Ma, X., 2023b. FAMG: A flow-aware and mixed granularity method for load-balancing in data center networks. Comput. Commun. 209, 415–428.

Majidi, A., Gao, X., Zhu, S., Jahanbakhsh, N., Zheng, J., Chen, G., 2023. MiFi: Bounded update to optimize network performance in software-defined data centers. IEEE/ACM Trans. Netw. 31 (1), 322–335.

Menikkumbura, D., Taheri, P., Vanini, E., Fahmy, S., Eugster, P., Edsall, T., 2023. Congestion control for datacenter networks: A control-theoretic approach. IEEE Trans. Parallel Distrib. Syst. 34 (5), 1682–1696.

Meza, J., Xu, T., Veeraraghavan, K., Mutlu, O., 2018. A large scale study of data center network reliability. In: Proceedings of the Internet Measurement Conference 2018. pp. 393–407.

Mohammed, M.A., Tăpuş, N., 2023. A novel approach of reducing energy consumption by utilizing big data analysis in mobile cloud computing. Mesop. J. Big Data 2023, 110–117.

Niu, M., Feng, Y., Sakurai, K., 2023. A two-stage detection system of ddos attacks in SDN using a trigger with multiple features and self-adaptive thresholds. In: 2023 17th International Conference on Ubiquitous Information Management and Communication. IMCOM, IEEE, pp. 1–8.

Noormohammadpour, M., Raghavendra, C.S., 2017. Datacenter traffic control: Understanding techniques and tradeoffs. IEEE Commun. Surv. Tutor. 20 (2), 1492–1525.
Nougnanke, B., Labit, Y., Bruyere, M., Aïvodji, U., Ferlin, S., 2023. ML-based performance modeling in SDN-enabled data center networks. IEEE Trans. Netw. Serv. Manag. 20 (1), 815–829.
Noura, M., Atiquzzaman, M., Gaedke, M., 2019. Interoperability in internet of things: Taxonomies and open challenges. Mobile Netw. Appl. 24, 796–809.
Pfaff, B., Lantz, B., Heller, B., et al., 2012. Openflow Switch Specification, Version 1.3. 0. Open Networking Foundation, pp. 39–46.
Pióro, M., Szentesi, Á., Harmatos, J., Jüttner, A., Gajowniczek, P., Kozdrowski, S., 2002. On open shortest path first related network optimisation problems. Perform. Eval. 48 (1–4), 201–223.
Priyadarsini, M., Bera, P., Das, S.K., Rahman, M.A., 2022. A security enforcement framework for SDN controller using game theoretic approach. IEEE Trans. Dependable Secure Comput. 20 (2), 1500–1515.
Qiu, T., Chen, N., Li, K., Atiquzzaman, M., Zhao, W., 2018. How can heterogeneous internet of things build our future: A survey. IEEE Commun. Surv. Tutor. 20 (3), 2011–2027.
Qiu, T., Chi, J., Zhou, X., Ning, Z., Atiquzzaman, M., Wu, D.O., 2020. Edge computing in industrial internet of things: Architecture, advances and challenges. IEEE Commun. Surv. Tutor. 22 (4), 2462–2488.
Qureshi, M.A., Cheng, Y., Yin, Q., Fu, Q., Kumar, G., Moshref, M., Yan, J., Jacobson, V., Wetherall, D., Kabbani, A., 2022. PLB: Congestion signals are simple and effective for network load balancing. In: Proceedings of the ACM SIGCOMM 2022 Conference. SIGCOMM '22, Association for Computing Machinery, New York, NY, USA, pp. 207–218.
Robin, D.D., Khan, J.I., 2022. P4TE: PISA switch based traffic engineering in fat-tree data center networks. Comput. Netw. 215, 109210.
Rojas-Cessa, R., Kaymak, Y., Dong, Z., 2015. Schemes for fast transmission of flows in data center networks. IEEE Commun. Surv. Tutor. 17 (3), 1391–1422.
Roy, A., Zeng, H., Bagga, J., Porter, G., Snoeren, A.C., 2015. Inside the social network's (datacenter) network. SIGCOMM Comput. Commun. Rev. 45 (4), 123–137.
Salhab, N., Rahim, R., Langar, R., 2020. Optimization of virtualization cost, processing power and network load of 5G software-defined data centers. IEEE Trans. Netw. Serv. Manag. 17 (3), 1542–1553.
Sen, S., Shue, D., Ihm, S., Freedman, M.J., 2013. Scalable, optimal flow routing in datacenters via local link balancing. In: Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies. pp. 151–162.
Shen, G., Li, Q., Shi, W., Han, F., Jiang, Y., Gu, L., 2022. Poche: A priority-based flow-aware in-network caching scheme in data center networks. IEEE Trans. Netw. Serv. Manag. 19 (4), 4491–4504.
Sherstinsky, A., 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D 404, 132306.
Sun, P., Guo, Z., Li, J., Xu, Y., Lan, J., Hu, Y., 2022. Enabling scalable routing in software-defined networks with deep reinforcement learning on critical nodes. IEEE/ACM Trans. Netw. 30 (2), 629–640.
Tan, L., Su, W., Liu, Y., Gao, X., Zhang, W., 2021. DCQUIC: Flexible and reliable software-defined data center transport. In: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops. INFOCOM WKSHPS, pp. 1–8.
Tang, Z., Zhang, T., Zhu, K., 2022. ORSM: Online routing and scheduling mechanism for mix-flows in data center networks. In: 2022 International Conference on Computer Communications and Networks. ICCCN, pp. 1–10.
Thaler, D., Hopps, C., 2000. Multipath Issues in Unicast and Multicast Next-Hop Selection. Tech. Rep.
Vanini, E., Pan, R., Alizadeh, M., Taheri, P., Edsall, T., 2017. Let it flow: Resilient asymmetric load balancing with flowlet switching. In: 14th USENIX Symposium on Networked Systems Design and Implementation. NSDI 17, pp. 407–420.
Wang, Z., Goudarzi, M., Gong, M., Buyya, R., 2024. Deep reinforcement learning-based scheduling for optimizing system load and response time in edge and fog computing environments. Future Gener. Comput. Syst. 152, 55–69.
Wang, Y., Li, Y., Wang, T., Liu, G., 2022. Towards an energy-efficient Data Center Network based on deep reinforcement learning. Comput. Netw. 210, 108939.
Wang, G., Ng, T.E., 2010. The impact of virtualization on network performance of amazon ec2 data center. In: 2010 Proceedings IEEE INFOCOM. IEEE, pp. 1–9.
Wang, B., Qi, Z., Ma, R., Guan, H., Vasilakos, A.V., 2015. A survey on data center networking for cloud computing. Comput. Netw. 91, 528–547.
Wang, J., Rao, S., Liu, Y., Sharma, P.K., Hu, J., 2023. Load balancing for heterogeneous traffic in datacenter networks. J. Netw. Comput. Appl. 217, 103692.
Wang, W., Sun, Y., Zheng, K., Kaafar, M.A., Li, D., Li, Z., 2014. Freeway: Adaptively isolating the elephant and mice flows on different transmission paths. In: 2014 IEEE 22nd International Conference on Network Protocols. IEEE, pp. 362–367.
Wei, Z., Li, Q., Zhu, K., Zhou, J., Zou, L., Jiang, Y., Xiao, X., 2023. DiffTREAT: Differentiated traffic scheduling based on RNN in data centers. IEEE Trans. Cloud Comput. 11 (3), 2407–2419.
Wilson, C., Ballani, H., Karagiannis, T., Rowtron, A., 2011. Better never than late: Meeting deadlines in datacenter networks. ACM SIGCOMM Comput. Commun. Rev. 41 (4), 50–61.
Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y., 2020. A comprehensive survey on graph neural networks. IEEE Tran. Neural Netw. Learn. Syst. 32 (1), 4–24.
Xia, W., Wen, Y., Foh, C.H., Niyato, D., Xie, H., 2014. A survey on software-defined networking. IEEE Commun. Surv. Tutor. 17 (1), 27–51.
Xia, Y., Wu, J., Xia, J., Wang, T., Mao, S., 2021. Multipath-aware TCP for data center traffic load-balancing. In: 2021 IEEE/ACM 29th International Symposium on Quality of Service. IWQOS, pp. 1–6.
Xia, W., Zhao, P., Wen, Y., Xie, H., 2016. A survey on data center networking (DCN): Infrastructure and operations. IEEE Commun. Surv. Tutor. 19 (1), 640–656.
Xu, Z., Lu, Y., Ma, X., 2021. BOBBLE: A mixed routing-granularity distributed load balancing for data center networks. In: 2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application.
Xu, Y., Mohammed, T., Di Francesco, M., Fischione, C., 2022. Distributed assignment with load balancing for DNN inference at the edge. IEEE Internet Things J. 10 (2), 1053–1065.
Zaher, M., Alawadi, A.H., Molnár, S., 2021. Sieve: A flow scheduling framework in SDN based data center networks. Comput. Commun. 171, 99–111.
Zainudin, A., Ahakonye, L.A.C., Akter, R., Kim, D.-S., Lee, J.-M., 2022. An efficient hybrid-dnn for ddos detection and classification in software-defined iiot networks. IEEE Internet Things J..
Zeng, G., Chen, L., Yi, B., Chen, K., 2022. Cutting tail latency in commodity datacenters with cloudburst. In: IEEE INFOCOM 2022 - IEEE Conference on Computer Communications. pp. 600–609.
Zhang, T., Huang, R., Hu, Y., Li, Y., Zou, S., Zhang, Q., Liu, X., Ruan, C., 2023. Load balancing with deadline-driven parallel data transmission in data center networks. IEEE Internet Things J. 10 (2), 1171–1191.
Zhang, H., Shi, X., Yin, X., Wang, J., Wang, Z., Guo, Y., Lan, T., 2021. Boosting bandwidth availability over inter-dc wan. In: Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies. pp. 297–312.
Zhang, J., Yu, F.R., Wang, S., Huang, T., Liu, Z., Liu, Y., 2018. Load balancing in data center networks: A survey. IEEE Commun. Surv. Tutor. 20 (3), 2324–2352.
Zhang, T., Zhang, Q., Lei, Y., Zou, S., Huang, J., Li, F., 2022. Load balancing with traffic isolation in data center networks. Future Gener. Comput. Syst. 127, 126–141.
Zhou, Y., Cheng, G., Yu, S., 2021. An SDN-enabled proactive defense framework for DDoS mitigation in IoT networks. IEEE Trans. Inf. Forensics Secur. 16, 5366–5380.
Zhou, J., Tewari, M., Zhu, M., Kabbani, A., Poutievski, L., Singh, A., Vahdat, A., 2014. WCMP: Weighted cost multipathing for improved fairness in data centers. In: Proceedings of the Ninth European Conference on Computer Systems. pp. 1–14.
Zou, S., Huang, J., Wang, J., He, T., 2021a. RMC: Reordering marking and coding for fine-grained load balancing in data centers. IEEE Trans. Comput. 69 (12), 8363–8374.
Zou, S., Huang, J., Wang, J., He, T., 2021b. Flow-aware adaptive pacing to mitigate TCP incast in data center networks. IEEE/ACM Trans. Netw. 29 (1), 134–147.

Yong Liu received Ph.D degree in Communication and Information System from Xidian University, Xi'an, China, in 2021. He is currently an associate professor at Hangzhou Normal University. His research interests include data center networks, software defined networking, and optical interconnected networks.

Tianyi Yu received the B.S. degree in Electronic and Information Engineering from Hangzhou City University, Hangzhou, China, in 2022. He is currently working toward the M.S. degree in Software Engineering with the School of Information Science and Technology, Hangzhou Normal University, Hangzhou, China. His research interests include data center networks and software defined networking.

Qian Meng received the MS. degree with the Department of Mathematics from Hangzhou Normal University, Hangzhou, China, in 2016, and the Ph.D degree with the Department of Telecommunication Engineering from Xidian University, Xi'an, China, in 2020. She is currently a Lecturer with the Department of Mathematics in Hangzhou Normal University. Her research interests include information security and applied cryptography.

Quanze Liu received the B.S. degree in Software Engineering from Chuanshan college, University of South China, Hengyang, China, in 2022. He is currently working toward the M.S. degree in Software Engineering with the School of Information Science and Technology, Hangzhou Normal University, Hangzhou, China. His research interests include data center networks and software defined networking.