Transmission Protocol for Video-on-Demand Streaming in 5G Core Networks

by

Si Yan

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Applied Science

in

Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2020

© Si Yan 2020

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The fifth generation (5G) networks are expected to accommodate various applications with diversified quality-of-service (QoS) requirements. To this end, some new technologies are employed in 5G networks such as software-defined networking (SDN), network function virtualization (NFV) and network slicing. SDN emerges as a promising architecture towards flexible and agile network operation. By decoupling control functions from substrate nodes, SDN enables informed network control and function reconfiguration on programmable switches. NFV partitions the network functions from the dedicated hardware platforms which enables on-demand deployment of network functions. Thanks to the employment of these new technologies, fine-grained and customized in-network control mechanisms can be realized to improve network performance.

In this thesis, we develop a transmission protocol for video-on-demand (VoD) streaming in the SDN/NFV-based 5G core network. By exploiting the flexibility of scalable video coding and the caching resources in the core network, we develop a novel selective caching policy for in-network congestion resolution. An enhanced transmission policy is proposed to improve the throughput and the network resource utilization by sending cached video packets once the congestion event is over. The proposed protocol is able to adapt to traffic dynamics and varying network environment, and it is shown to effectively alleviate network congestion with balanced throughput and resource utilization. Simulation results are presented to validate the efficiency of the proposed protocol in terms of packet delay, goodput ratio, throughput and resource utilization.

Acknowledgements

First and foremost, I would like to express my sincere appreciation to my supervisor, Professor Weihua Zhuang, for her continuous help and support during this M.A.Sc study. Her valuable advice, generous knowledge sharing and encouragement make this thesis possible. Professor Zhuang is always patient and helpful during the research discussion. Her professional guidance leads me to understand how to conduct in-depth research. It is a great honour for me to be one of her students.

I wish to express my gratitude to Professor Xuemin Shen for the organization of weekly group meeting. Professor Shen gives the invaluable instructions for both academic research and personal life. Professor Shen also helps us to understand the significance of caring our colleagues, friends and family. His advice will always play an important role in my life. In addition, I wish to thank Professor Xuemin Shen and Professor Zhou Wang for serving my thesis readers.

I would like to express my special thanks to Dr. Qiang Ye for his great support. He is helpful and always available to discuss research issues. I would also like to thank Kaige Qu for providing important comments for this research. I wish to extend my sincere gratitude to all the BBCR members, Dr. Peng Yang, Dr. Ning Zhang, Jiayin Chen, Dr. Junling Li, Dr. Weisen Shi, Dr. Omar Alhussein, Dr. Wei Quan, Dr. Phu Thinh Do, Dr. Wen Wu, Dr. Jianbing Ni, Dr. Nan Cheng, Dr. Yujie Tang, Mushu Li, Dongxiao Liu, Cheng Huang, Kangjia Lyu, Haixia Peng, A-Long Jin, Huaqing Wu, Liang Xue, Xuehan Ye, Conghao Zhou, Dr. Nan Chen, Dr. Jie Gao, Dr. Qihao Li, Dr. Yuan Zhang, and many others. My gratitude also goes to my friends, Dr. Biyuan Li, Zhaoyi Wang, Yu Wu, Baoguo Niu,

Zhangmao Wang.

Finally, I would like to thank my family for the great help and selfless support, my mother Chen Tang, my father Haiqing Yan and my wife Yajing Li.

Dedication

This thesis is dedicated to my parents, Chen Tang and Haiqing Yan, and my wife, Yajing Li.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

4G	Fourth-Generation
5G	Fifth-Generation
ACK	Acknowledgment
ADF	Augmented Dickey-Fuller Test
AICC	Corrected Akaike Information Criterion
AIMD	Additive-Increase Multiplicative-Decrease
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
BDP	Bandwidth-Delay Product
BIC	Binary Increase Control
CDF	Cumulative Distribution Function
CUPS	Control and User Plane Separation
CWND	Congestion Window
DCCP	Datagram Congestion Control Protocol

E2E End-to-End

FIFO First-In-First-Out

HTTP Hypertext Transfer Protocol

IDS Intrusion Detection System

IP Internet Protocol

MAB Multi-Armed Bandit

MLE Maximum Likelihood Estimation

NFV Network Function Virtualization

QoS Quality-of-Service

QUIC Quick UDP Internet Connections

RTO Retransmission Timeout

RTT Round-Trip Delay Time

SCTP Stream Control Transmission Protocol

SDN Software-Defined Networking

SDRA Software-Defined Resource Allocation

SDT Software-Defined Topology

SFC Service Function Chain

SVC Scalable Video Coding

TCP Transmission Control Protocol

UDP User Data Protocol

UPF User Plane Function

VNF Virtual Network Function

VoD Video-on-Demand

List of Symbols

Δ_s	Length of a video segment
$\lambda_j^{(l)}$	Arrival rate of the j-th cross-traffic flow at V_l
$\hat{\tau}(z)$	Sample autocovariance function
$\mathcal{A}$	Set of all possible action tuples
$a(k)$	Control action for the k-th time slot
$a_1(k)$	Action for selective caching functionality
$a_2(k)$	Action for enhanced transmission functionality
C_l	Total capacity of V_l
d	Degree of differencing
$\boldsymbol{D_a}$	Matrix of observations for arm a
$d_a(k)$	Average E2E delay of the k-th time slot
$\boldsymbol{I_d}$	The $d \times d$ identity matrix
$\mathcal{I}_l$	Set of traffic flows at V_l
M_l	Number of cross-traffic flows at node V_l

N_e	Number of enhancement layers of a video segment
N_E	Maximum number of ET-chunks in a time slot
N_v	Number of video clients belong to the VoD streaming slice
p	Order of the autoregressive model
q	Order of the moving-average model
$r(k)$	E2E available capacity of VoD streaming slice for the k-th time slot
$\boldsymbol{R_a}$	Response vector for arm a
$R_{a(k)}(k)$	Reward of executing action $a(k)$ in the k-th time slot
r_e	E2E available capacity of VoD streaming slice
r_l	Available capacity of V_l
$\hat{t}(k)$	Predicted video traffic load in the k-th time slot
$t_i(k)$	Observed Video traffic load of layer i in the k-th time slot
$\hat{t}_i(k)$	Predicted video traffic load of layer i in the k-th time slot
T_r	E2E delay requirement
T_s	Duration of a time slot
V_l	The l-th node in the VoD streaming slice
$\boldsymbol{x_{k,a(k)}}$	Context information of the k-th time slot

Chapter 1

Introduction

To satisfy the diverse demands posed by different services, 5G networks are required to be flexible and programmable. Core networks play an important role in the 5G systems, which are located between the access networks and the data networks. The traffic of same service from different end hosts is aggregated at the edge node of the core network and is processed by certain network functions such as firewall and intrusion detection system (IDS) [1]. Different with the core network of fourth generation (4G) networks, 5G core networks employ SDN, NFV and network slicing technologies to fulfil more stringent service requirements. With SDN, the network control intelligence is separated from the user plane [2]. NFV decouples the network functions from the dedicated hardware platforms, which enables the flexible deployment of the softwarized network functions. Given the transmission path of a traffic flow in the core network, a set of resources along the path are allocated to the flow by network slicing. To better operate the network, the traffic load of each flow in the core network should be well controlled. Thus, a properly designed

transmission protocol is required to adjust the flow traffic load according to different network conditions. A large number of traffic flows from different services traverse the same core network which have heterogeneous QoS requirements. Therefore, the transmission protocol should be customized for each type of service. In this research, we aim to develop a customized transmission protocol for VoD streaming service. In this chapter, we first discuss the architecture and the characteristics of the 5G core network. Then, the literature survey on the existing transmission protocols is presented. At last, we introduce the research contributions and the organization of this thesis.

1.1 5G Core Networks

Currently, the telecom industry is evolving from the 4G system to the 5G system [3]. A significant change of the core network from 4G to 5G systems is the control and user plane separation (CUPS). By employing SDN, the control plane is decoupled from the data plane. Note that the phrases user plane and data plane are used interchangeable throughout the thesis. The control plane is a centralized controller which collects the global information of the network to manage the traffic flows traversing the core network. The main responsibilities of the data plane are packet forwarding, packet inspection and QoS management. A typical architecture of the 5G core network is shown in Fig. 1.1. Nine network functions are implemented in the control plane to manage the network operations. The roles of these network functions are introduced in [4]. The horizontal line between the network functions in the control plane is a bus which realizes the communication between network functions. The user plane function (UPF) is the data plane of the core

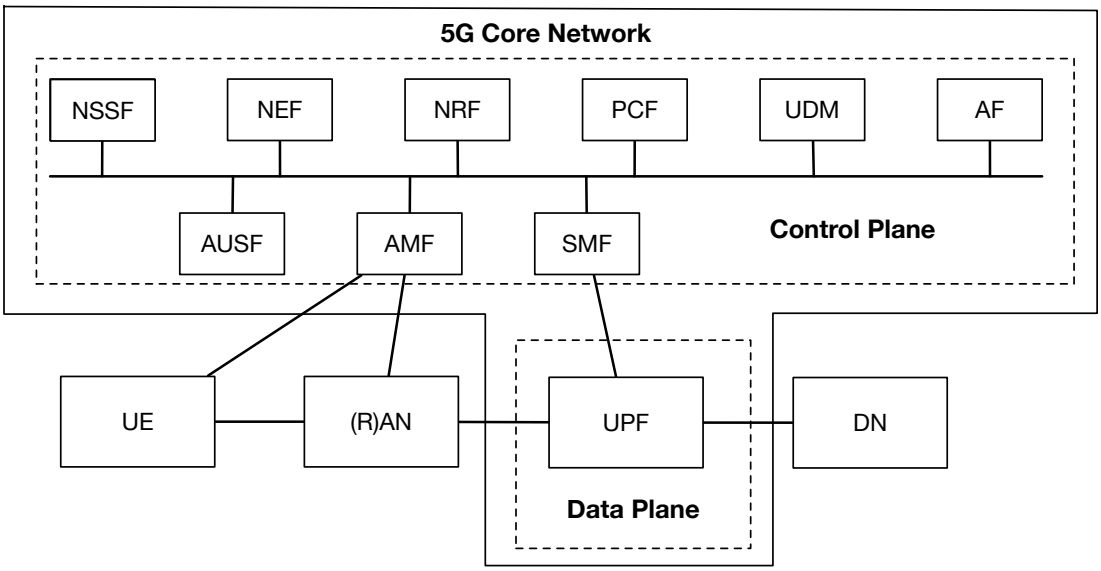

Figure 1.1: An illustration of the 5G system.

network which achieves the connectivity between access networks and data networks (e.g., Internet). When a traffic flow passes through the UPF, it is processed by a sequence of network functions (e.g., firewall) to fulfil the service requirements. The line which connects the SMF to the UPF represents the southbound interface between the control plane and the data plane. OpenFlow protocol [3] is widely used in the industry to manage the signalling in the southbound interface. Except for SDN, the 5G core network also adopts the NFV technology which decouples the network functions from the underlying hardware [5]. In the 4G core network, network functions are installed on proprietary platforms. If the resources or the functionalities of the platform are not enough for the service requirements, network operators have to replace a new platform to cater the demands. With NFV, network functions in the core network are softwarized and implemented in virtual machines which can be installed in commodity servers [6]. The softwarized network functions are referred

to as virtual network functions (VNFs). If the resources of a commodity server cannot afford all the VNFs running on it, some VNFs are migrated to other servers which have more idle resources. By doing so, the VNFs can be flexibly embedded onto the servers deployed in the core network [7]. The commodity servers which host VNFs are called NFV nodes.

The UPF (i.e., data plane of the 5G core network) has three main components, i.e., in-network switches, physical links and NFV nodes. As mentioned, each traffic flow is processed by certain VNFs when it passes through the UPF. The chain of these VNFs is also known as a service function chain (SFC). In-network switches and physical links are located between two NFV nodes for packet forwarding. Determining the transmission path of each traffic flow in the UPF is an essential issue for 5G system development which is referred to as software-defined topology (SDT) [7, 8]. The SDT algorithm is deployed on the control plane and makes decisions based on the global view of the UPF. Given the transmission paths of the traffic flows in the UPF, the resources of a network element is required to be properly allocated to flows traversing this network element. The resource allocation problem in an SDN/NFV-enabled 5G core network is called software-defined resource allocation (SDRA) [8]. SDT and SDRA mechanisms control the traffic flows and the UPF at a large time scale (e.g., several hours) which are not suitable to manage the short-term traffic burstiness. More fine-grained transmission protocols are required to manage the traffic flows from different services at smaller time scales.

1.2 Transmission Protocols

Previously, the main effort of transmission protocol development is put into the improvement of transport layer protocols at end hosts which adjust source sending rates to keep the balance between packet E2E delay and throughput. Transmission control protocol (TCP) and its variants are the most popular transport layer protocols [9, 10, 11, 12, 13, 14, 15]. Some brand new protocols are also proposed such as quick UDP Internet Connections (QUIC) [16, 17], stream control transmission protocol (SCTP) [18] and datagram congestion control protocol (DCCP) [19]. The motivation of developing new transport layer protocols is to eliminate some limitations of TCP. However, the deployment progress of these protocols is slow. The main reason is the lack of support by middleboxes [9] which are widely implemented in the network. The middleboxes always reject non-TCP or non-UDP traffic and drop the corresponding packets. Although QUIC mitigates this issue by designing the protocol upon the user data protocol (UDP), its deployment is still limited since the protocol requires to modify the current end hosts which increases the deployment costs. Thus, the literature survey in this section focuses on the discussion on TCP and its variants, which can be classified into three categories, i.e., loss-based protocols, delay-based protocols and capacity-based protocols [9]. A TCP-like transport layer protocol maintains a congestion window (CWND) at the source node to adjust the source sending rate based on the observations of the packet round-trip delay time (RTT). The RTT is defined as the time between the instant source node sends a packet and the instant the acknowledge (ACK) packet arrives at the source node. Additive-increase multiplicative-decrease (AIMD) is one of the most popular CWND update mechanisms. The main principle of

CWND update mechanisms is that increasing the CWND if no network congestion on the path and decreasing the CWND when a congestion event is detected. The existing protocols use different congestion indicators such as packet loss, packet delay and network capacity.

1.2.1 Loss-Based Protocols

Loss-based protocols are the major transport layer protocols employed by the end hosts in the network [9] which reduce the CWND if a packet is lost during the transmission. When a congestion event occurs at a node on the transmission path, the queue length of the buffer at the congested node becomes larger. Once the buffer is full, the incoming packets are dropped by the congested node and a packet loss event is created. The sender regards two phenomenon as an indicator of packet loss events, namely, 1) the sender does not receive the ACK for a transmitted packet before the required deadline (i.e., retransmission timeout timer expires), and 2) the sender receives three duplicate ACKs. The first phenomenon represents more severe network congestion since the sender cannot receive ACK packet before the retransmission timeout (RTO). However, the receipt of duplicate ACKs can be triggered if a packet loss event occurs or the packet arrivals at the receiver is out of order.

TCP Tahoe reduces the CWND to 1 as long as a packet loss is detected which has negative impact on the throughput if the disordered packet arrivals trigger the duplicate ACKs [20]. To limit the throughput reduction, the CWND of TCP Reno [20] and TCP NewReno [21] is reduced to its half when the packet loss event is indicated by three duplicate ACKs. The study in [9] shows that these three transport layer protocols are less efficient

when the bandwidth-delay product (BDP) of the transmission path becomes larger. Binary increase control (BIC) [22] is proposed to replace the AIMD mechanism for the networks with large BDP. Although BIC improves the QoS performance, it is too aggressive to keep the fairness with other traffic flows [23]. The fairness issue of BIC is resolved by its enhanced version, i.e., CUBIC [24], which utilizes a cubic function to control the size of CWND. The CWND of CUBIC increases fast when it is small. However, the increasing speed of CWND becomes slow when the CWND is getting larger. By doing so, the fairness between the CUBIC flows and other TCP flows are kept. Even though CUBIC achieves good performance and fairness simultaneously, it still suffers a common issue for the loss-based protocols, i.e., bufferbloat [25]. In recent years, the buffer at the intermediate nodes in the network is getting larger which means that it is difficult to fill the buffer even during the network congestion. As a result, the sender fails to sense the congestion event since no packet loss event is detected and keeps increasing the sending rate which exacerbates the network congestion and degrades the QoS performance. This phenomenon is referred to as bufferbloat.

1.2.2 Delay-Based Protocols

To mitigate the QoS degradation caused by bufferbloat, delay-based transport layer protocols are proposed which use the RTT of the packets to control the source sending rate. TCP Vegas [11] and its enhanced versions [26, 27, 28, 29] are broadly studied in this category. The main idea of these protocols is to trigger the congestion control mechanisms (i.e., reduce the CWND) at the sender as soon as the measured RTT is greater than the

threshold instead of waiting the first packet loss occurs in the network. Thus, the packet delay is well controlled at the early stage of a congestion event. TCP Verus [30] is another delay-based protocol which adjusts the CWND based on delay variations. It is discussed in [9] that TCP Verus requires much computing resources which impedes its large-scale deployment. To reduce the signalling overhead, the receiver of some TCP-like protocols employs the delayed ACK scheme which leads to inaccurate measurement of the RTT at the sender. The inaccurate RTT may degrade the performance of some delay-based protocols. Although the protocols discussed in this subsection solve the bufferbloat problem, the deployment of delay-based protocols is still limited [9] which caused by the fairness issue when the flows of delay-based protocols and loss-based protocols traverse the same congested node with large buffer. The sender with delay-based protocols reduces its sending rate once the RTT increases to a certain value. In the contrary, the sender with loss-based protocols consistently increases the CWND until the first packet loss event is detected. As a result, the available resources at the congested node are gradually occupied by the flows of loss-based protocols. Considering the fact that loss-based mechanisms dominate the transport layer protocols implemented in the Internet [9], large-scale deployment of delay-based protocols still needs more effort.

1.2.3 Capacity-Based Protocols

Capacity-based protocols adjust the source sending rate based on the estimation of the transmission path bandwidth [31]. Westwood is a TCP-like capacity-based protocol [12]. It can achieve better performance than loss-based protocols if the transmission path has

high packet loss rate which is not caused by congestion (e.g., link error) [9]. The sender with loss-based protocols reduces its packet sending rate whenever it detects a packet loss. As a result, the throughput of loss-based protocols is degraded. However, TCP Westwood adjusts its CWND to adapt to the network available capacity which overcomes the above-mentioned throughput reduction.

1.3 Motivation and Research Contributions

As discussed in Section 1.2, massive end hosts implement loss-based protocols to control their sending behaviours. However, loss-based protocols fail to make timely reaction to the congestion event since the bufferbloat issue. If a congestion event occurs in the core network, the senders with loss-based protocols keep increasing their sending rate during the network congestion until the first packet loss event is detected. The packet delay of all the traffic flows which pass through the congested network element increases rapidly. The main reason for the bufferbloat issue of loss-based protocols is that sender does not have enough information of the network dynamics. This limitation exists in many E2E protocols which are deployed at the end hosts to control the source sending rate based on the network feedbacks such as RTT. Except for the delayed reaction to the network congestion, loss-based protocols also prevent the deployment of other transport layer protocols such as delay-based protocols. Since the core network plays an important role in the 5G systems, we aim to develop a transmission protocol in the 5G core network which achieves in-network control for the flows from the senders with loss-based transport layer protocols. The proposed protocol is required to make fast reaction to the congestion events in the

core network, i.e., the packet delay in the core network should be controlled in a certain range during the network congestion. The traffic of the same service from different senders formulates one aggregated traffic flow whose packets traverse the same path in the data plane of the core network. In the rest of this thesis, the phrase traffic flows indicates the aggregated traffic flows in the core network. Also, the transmission protocol is proposed to control the aggregated traffic flows.

The traffic flows of different services pass through the same core network which pose various service requirements. To better control the traffic flows, the proposed transmission protocol should be customized and service-oriented to achieve differentiated QoS provisioning. VoD streaming service is one of the most important services for 5G networks [32]. Currently, TCP-like protocols become to the major transport layer protocols for video delivery [33, 34]. In this research, we intend to develop a customized transmission protocol in the core network to manage the video traffic flows. The proposed protocol is referred to as SDP-VS. The research contributions of this thesis are summarized as follows:

1. Since caching resources are employed in 5G core networks [35], we design a selective caching functionality for SDP-VS which temporarily puts certain video packets to the caching buffer during the network congestion and keeps the balance between the packet delay and throughput;

2. We develop an enhanced transmission functionality for SDP-VS that transmits the cached packets to the corresponding video clients once the network condition improves. With enhanced transmission functionality, the utilization of the available resources and the throughput of video traffic flows are enhanced.

1.4 Organization of The Thesis

The rest of this thesis is organized as follows. In Chapter 2, the system model under consideration and the research problems are discussed. Chapter 3 presents the proposed SDP-VS protocol which includes a detailed description of the protocol operation, the required traffic prediction algorithm, and the mechanism of selecting control actions via machine learning technology. Simulation results are discussed in Chapter 4, which demonstrate the efficiency of the proposed SDP-VS protocol. Finally, Chapter 5 concludes this research and provides several future research directions.

Chapter 2

System Model and Problem Statement

2.1 System Model

2.1.1 Network Model

We focus on a 5G core network where traffic of the same service from different source nodes is aggregated as one traffic flow at the edge node. As shown in Fig. 2.1, multiple traffic flows traverse the core network. To satisfy the service requirements, each traffic flow is required to be processed by a chain of VNFs which are implemented on the NFV nodes in the core network. We assume that only one VNF is installed on an NFV node. The traffic flow is forwarded by in-network switches over physical links between two NFV nodes. The transmission path of each traffic flow in the core network is determined by the

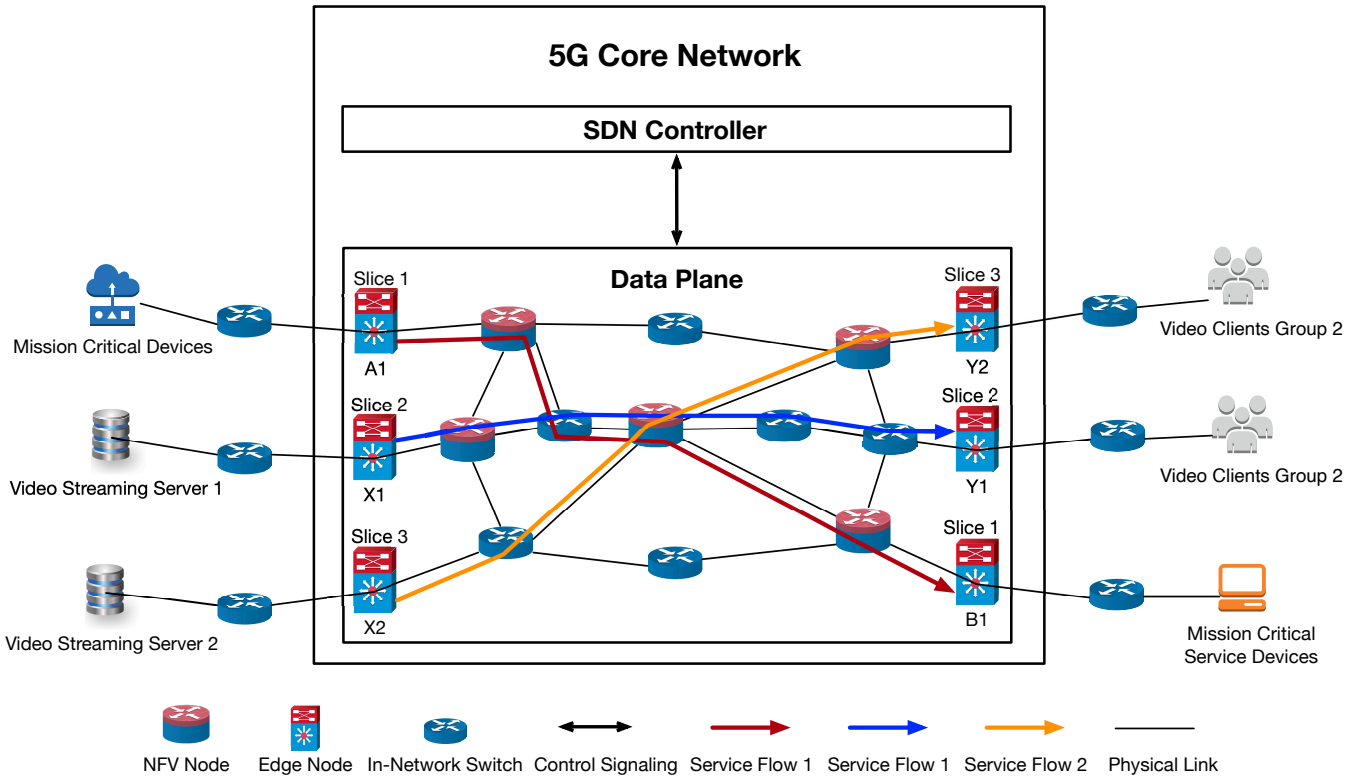

Figure 2.1: Multiple services network topology.

SDN controller [36]. The SDN controller collects the global information of the network and determines the path for each traffic flow in the core network. To improve resource utilization, multiple traffic flows pass a common network element (in-network switch, physical link or NFV node) and share the same set of resources [37]. Two types of resources are considered in the 5G core networks, i.e., 1) the computing resources at NFV nodes, and 2) the transmission resources at in-network switches and over physical links [1]. Given the transmission path and the allocated resources of a traffic flow, a customized transmission protocol is deployed in the core network for this flow to achieve service-oriented control. For each traffic flow in the core network, we define a *slice* which includes a transmission

path with allocated resources and customized transmission protocol.

VoD Streaming Slice

A unicast VoD streaming slice has a linear topology between a pair of edge nodes (e.g., Slice 2 in Fig. 2.1). The set of nodes in the slice is denoted by $\mathcal{V} = \{V_1, V_2, \ldots, V_L\}$. A node is either an in-network switch or an NFV node which has an unlimited first-in-first-out (FIFO) queue to buffer the arrived packets of the traffic flows passing through. The bottleneck resource type of an in-network switch (NFV node) is the transmission resources (computing resources). In the rest of this thesis, the resource type of a node in the VoD streaming slice refers to the corresponding bottleneck resource type. At each node, the video traffic flow shares the resources with multiple cross-traffic flows. The number of cross-traffic flows traversing node V_l is denoted by M_l. The set of traffic flows at V_l, denoted by $\mathcal{I}_l$, consists of one video traffic flow of interest and M_l cross-traffic flows. The traffic arrivals of a cross-traffic flow can be represented by a series of non-overlapping traffic segments [8]. Within a traffic segment, the traffic statistics such as arrival rate are stationary. The time instant when the traffic statistics start to change is referred to as a traffic change point. At time instant t, $\lambda_j^{(l)}(t)(j = 1, 2, \ldots, M_l)$ represents the corresponding arrival rate of the j-th cross-traffic flow at V_l. If two consecutive change points are detected at time instants T_1 and T_2, $\lambda_j^{(l)}(t)$ is constant when t is in $[T_1, T_2]$. Denote by C_l the total capacity in packet/s of V_l. We define the available capacity, $r_l(t)(l = 1, 2, \ldots, L)$, of an in-network switch (NFV node) as the transmission resources (computing resources) in packet/s left

over by cross-traffic flows [38], given by

$$r_l(t) = C_l - \sum_{j=1}^{M_l} \lambda_j^{(l)}(t). \tag{2.1}$$

The E2E available capacity, $r_e(t)$, of a VoD streaming slice is determined by the bottleneck node which has the minimum available capacity, i.e., $r_e(t) = \min\{r_1(t), r_2(t), \ldots, r_L(t)\}$.

The server-side edge node (client-side edge node) of the VoD streaming slice is the ingress node (egress node) which is assumed to have unlimited caching resources. For example, nodes X_1 and Y_1 in Fig. 2.1 are the ingress and egress nodes of Slice 2. For backward compatibility on end hosts, the ingress (egress) node is an in-network proxy server which maintains the TCP connections with the video server (clients) [39]. The ingress node replies an ACK packet to the video server for every received video packet. For all the video packets received by the egress node, their payload is extracted and encapsulated into new TCP packets. The TCP packets are copied and cached at the egress node, and are then forwarded to the corresponding video clients. Let N_v denote the number of video clients associated to a VoD streaming slice. The video client replies an ACK packet for each received video packet for acknowledgement. When the egress node receives an ACK packet from the video client, it removes the corresponding video packet from the egress node caching buffer. However, if a video packet is lost between the egress node and the video client, the egress node either receives three duplicate ACKs of the previous packet or experiences retransmission timeout. In this case, the egress node retransmits the lost packet and activates the TCP congestion control mechanism.

2.1.2 VoD Streaming System

The scalable video coding (SVC) technique is used to encode video files in the video server [40]. Each video is divided into a series of video segments. Denote by Δ_s the length of a segment. Each segment is further encoded into several layers, including one Base Layer and N_e Enhancement Layers. Different layers of a video segment can be stored and streamed independently in the form of small video chunks. The base-layer chunks are necessary to decode segments at video clients. An enhancement-layer chunk can be decoded only if all the lower enhancement-layer chunks and the base-layer chunk from the same segment are received completely by the client. The more enhancement-layer chunks are received, the higher video quality will be. Before sending the chunks into the network, each chunk is fragmented and encapsulated into multiple video packets. The quality (i.e., the number of SVC layers) of the streamed segments is controlled by the video clients [33, 41]. When all the base-layer packets of the requested segments are received by a client, the client needs to determine the quality of the following several segments based on the current buffer level, i.e., the number of playable video segments in the client buffer. The desired quality information is transmitted to the video server by the HTTP GET message.

2.1.3 Protocol Functionalities

To achieve in-network control for VoD streaming service, SDP-VS incorporates the following functionalities: header conversion functionality, selective caching functionality, and enhanced transmission functionality [42]. When a congestion event occurs in the VoD streaming slice, the ingress node selectively put incoming packets into the caching buffer.

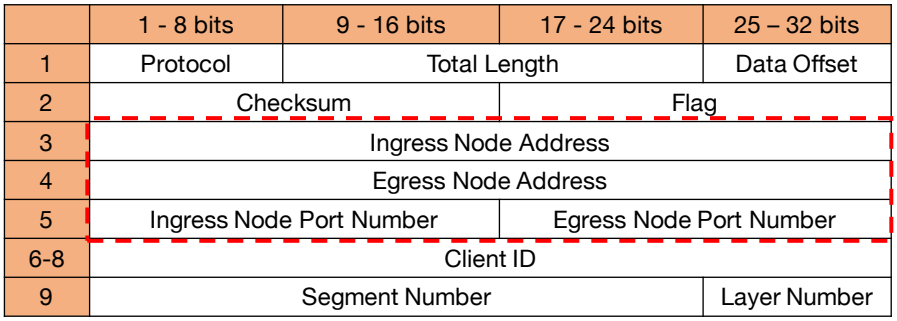

Figure 2.2: The SDP-VS header.

Once the network condition improves, packets that help to enhance video quality can be retrieved from the caching buffer for enhanced transmission. Time is partitioned into time slots of constant duration T_s [43]. At the beginning of each time slot, the ingress node of VoD streaming slice selects appropriate functionality based on the network condition. A description of the protocol functionalities of SDP-VS is presented as following:

1. **Header conversion functionality:** It is deployed at the ingress node to add SDP-VS header over all the video packets. The SDP-VS header format is shown in Fig. 2.2. Between the edge nodes of a VoD streaming slice, the source/destination address of the video packet is the *Ingress/Egress Node Address*. The sending/receiving port number at the ingress/egress node is presented in the *Ingress/Egress Node Port Number* field. The fields in the red dashed block is referred to as a slice ID for slice differentiation. The *Protocol* field indicates the applied transmission protocol for the video traffic flows in the core network, i.e., SDP-VS. The *Total Length*, *Data Offset* and *Checksum* fields are required by all the IP packets in the network. The *Flag* field is used to differentiate the types of packets in the VoD streaming slice. The

Client ID, *Segment Number* and *Layer Number* fields are required by the proposed SDP-VS. The client ID contains the IP addresses and port numbers of the server and clients. The segment number and layer number of a video packet are extracted from the application layer payload by the ingress node, and are then added to the corresponding fields. Note that the layer number of base-layer packets is 0 and i-th enhancement-layer packets is i;

2. **Selective caching functionality:** SVC codec enables flexible video decoding, and video contents can be successfully decoded even in the absence of enhancement layer packets. Hence, higher layer packets can be selectively cached in the network, without significant degradation of user experience. By exploiting the caching resources, instead of dropping packets when network is congested, we design a selective caching policy to temporarily store certain packets on the ingress node, which benefits fast response to network dynamics;

3. **Enhanced transmission functionality:** To compensate video quality once network condition improves, we design the enhanced transmission functionality for SDP-VS. If the enhanced transmission functionality is activated in a time slot, the ingress node is required to decide how many cached packets should be transmitted in this time slot. Then, the cached packets are pushed from the caching buffer to the VoD streaming slice.

2.1.4 Performance Metrics

To verify the performance improvement by deploying in-network control for VoD streaming service, we compare four types of QoS performance of the VoD streaming systems with and without the proposed in-network control. The QoS measures are:

1. **Average E2E delay (second)**, which is the average delay among the packets left the egress node of VoD streaming slice during a time slot. The E2E delay only includes the queueing delay, processing delay, transmission delay and propagation delay in the core network;

2. **Throughput (packet/s)**, which is the number of video packets that pass through the egress node of VoD streaming slice in one second. The throughput of a time slot is measured by the egress node which is equal to the total number of packets pass through the egress node during this slot over the length of the time slot;

3. **Goodput ratio**, which is the ratio of the number of packets whose E2E delay is less than required delay bound T_r over the total number of packets pass through the egress node of VoD streaming slice during a time slot;

4. **Resource utilization**, which is the ratio of throughput over E2E available capacity of VoD streaming slice.

2.2 Research Problem

Currently, E2E transmission protocols (e.g., TCP) are implemented at the end hosts to react to the time-varying network conditions. The sender adjusts its sending rate to balance the trade-off between packet E2E delay and throughput. However, E2E protocols only have limited network information (e.g., per-packet RTT) to control the sending behaviours. Moreover, delayed network statistics may lead to improper decisions. It has been shown that adding in-network control for traffic flows can enhance their QoS performance [37, 44], since a properly designed in-network control mechanism guides the network elements (e.g., in-network switches or NFV nodes) to have appropriate reactions to network environment dynamics. In packet-switching networks, packets of one traffic flow may traverse different paths to the destination. As a consequence, the network environment faced by each packet may be different. Hence, it is difficult to deploy a flow-level in-network control which takes all the related network conditions into consideration. The SDN technique has been widely studied for the 5G core network [37, 1]. As introduced in Section 2.1, the centralized controller of SDN collects global network information and determines the path of every traffic flow in the network [45]. With SDN, packets from one traffic flow pass through the same network path, which is helpful for designing the flow-level in-network control since the network conditions have similar impact on the packets of a single traffic flow.

A large number of traffic flows from multiple types of services traverse the same core network. Different services have diverse QoS requirements, which requires service-oriented control mechanisms for differentiated QoS provisioning. Considering VoD streaming service as a typical service in 5G networks [32], we intend to design a customized in-network con-

trol scheme for video traffic flows in the 5G core network. This research problem contains two main subproblems: 1) how to determine an in-network congestion management mechanism to mitigate QoS degradation caused by network congestion, and 2) how to design a throughput enhancement scheme under the condition that the network congestion level is well controlled. In this research, we propose an SDP-VS protocol deployed in the VoD streaming slice to realize customized in-network control for video traffic flows. The SDP-VS is composed of two functionalities, i.e., selective caching and enhanced transmission.

To mitigate network congestion in the VoD streaming slice, the ingress node caches some enhancement-layer packets through selective caching functionality. Caching insufficient video packets results in a large E2E delay. However, removing excess packets from the video traffic flow is harmful to the throughput performance. To better support the video delivery, selective caching functionality is required to determine an appropriate number of video packets which should be cached to keep the balance between average E2E delay and throughput. Two network factors are necessary in the decision making, i.e., video traffic load and E2E available capacity of the VoD streaming slice. Since the decision of selective caching functionality is made at the beginning of each time slot, a traffic prediction module needs to be identified. When the congestion event is over and more resources are available for the VoD streaming slice, the enhanced transmission functionality is activated to enhance the throughput and network resource utilization. To better use the network resources without generating new congestion event, we need to investigate the rules of executing the enhanced transmission functionality, i.e., determining how many cached packets should be transmitted in each time slot. Similar to selective caching functionality, video traffic load and E2E available capacity factors are considered in the design of enhanced transmission

functionality.

To determine the control actions (i.e., selective caching and enhanced transmission) for different network conditions, a model that captures the relationship between QoS and network condition is required. The network conditions include video traffic load and E2E available capacity of the VoD streaming slice. The proposed SDP-VS is expected to handle the real-world traffic which may not follow a known process (e.g., Poisson process). In addition, the E2E available capacity is influenced by all the cross-traffic of the VoD streaming slice which results in high complexity in mathematically characterizing the E2E available capacity. Considering the difficulty of building the analytical model, we formulate the action-selection problem as a multi-armed bandit (MAB) problem to maximize the expected overall performance. The ingress node selects the control action based on exploration-exploitation mechanism and observes the corresponding reward at the end of each time slot. Then, the action-selection strategy is updated based on the observed reward.

The proposed SDP-VS protocol is presented in Chapter 3.

2.3 Summary

In this chapter, we introduce the system model which includes network model, VoD streaming system and the performance metrics used to evaluate the proposed transmission protocol. Based on the system model, we identify the research problem for this research.

Chapter 3

A Learning-Based Software-Defined Protocol for VoD Streaming Service

The proposed SDP-VS protocol is presented in this chapter. First, we describe the framework of operating the proposed protocol which includes three main components, i.e., 1) traffic prediction module, 2) E2E available capacity measurement module, and 3) machine learning module. Then, a detailed description of the traffic prediction algorithm is given. At last, we present the strategy of selecting control actions via MAB learning.

3.1 SDP-VS Framework

SDP-VS controls the packet queueing delay during the network congestion and enhances the throughput once the congestion event is over by adjusting the traffic load for VoD

streaming slice. It achieves traffic management by executing different control actions at the ingress node. When the selective caching functionality is activated, some incoming video packets are removed from the video traffic flow and are cached in the caching buffer at the ingress node. If the enhanced transmission functionality is required, the corresponding video packets are transmitted from the ingress node to the video clients. To operate SDP-VS, three functional modules are implemented at the ingress node of VoD streaming slice, i.e., machine learning module, video traffic prediction module and E2E available capacity measurement module. The relationship among the modules is presented in Fig. 3.1. The machine learning module is the core of SDP-VS which selects the control action for each time slot based on the output of the other two functional modules. The video traffic prediction module estimates the traffic load of the next time slot based on the traffic loads observed in the last several time slots. The E2E available capacity measurement module is used to monitor the available capacity for the VoD streaming slice during the network operation.

Denote by $\hat{t}(k)$ and $r(k)$ the output of video traffic prediction module and E2E available capacity measurement module for the k-th time slot, respectively. If the k-th time slot starts at the time instant t_k, $r(k)$ is equal to $r_e(t_k)$. Considering the machine learning module selects action $a(k)$ for the k-th time slot which is represented as a two-tuple $(a_1(k), a_2(k))$. The first element of $a(k)$ indicates the action of selective caching functionality. In the k-th time slot, the ingress node caches all the incoming packets whose layer number is greater than $a_1(k)$. To avoid of creating a video rebuffering event, the base-layer packets are not considered in selective caching. When $a_1(k)$ is equal to 0, all the enhancement-layer packets arrived at the ingress node during the k-th time slot are

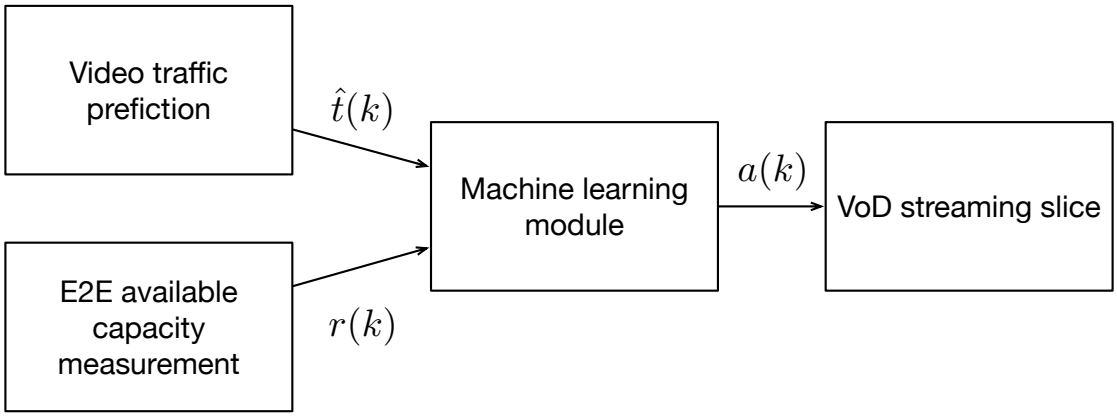

Figure 3.1: The framework of SDP-VS.

pushed to the caching buffer at the ingress node. If $a_1(k)$ equals N_e, no video packet needs to be cached in the k-th time slot. The second element of the action tuple denotes the action of enhanced transmission functionality. To reduce the size of action space, $a_2(k)$ is the number of chunks which should be transmitted by enhanced transmission functionality in the k-th time slot. In order to differentiate between the chunks in protocol operations and the video chunks discussed in Subsection 2.1.2, we denote the chunks in enhanced transmission by ET-chunks. All the ET-chunks contain the same number of video packets. Denote by N_E the pre-determined maximum number of ET-chunks transmitted in one time slot. Let $\mathcal{A}$ denote the set of all possible action tuples (i, j) where $i = 0, 1, \ldots, N_e$ and $j = 0, 1, \ldots, N_E$.

Here, we present an example to explain the reason why selective caching and enhanced transmission functionalities may coexist in one time slot. Suppose a video file is encoded into five SVC layers and the packets of each layer arrive at the ingress node of VoD streaming slice with a constant rate 1000 packet/s, and the E2E available capacity is 2200 packet/s.

If the action of selective caching functionality is 1, then all the incoming video packets whose layer number is greater than 1 are cached at the ingress node. As a result, only the video packets of base layer and layer 1 traverse the VoD streaming slice. Then, the arrival rate of the packets from the lowest two layers is 2000 packet/s. Considering the available capacity is greater than the traffic arrival rate, the available resources of the VoD streaming slice is not fully utilized. Sending a proper number of cached video packets in this time slot can increase the resource utilization without leading to network congestion.

3.1.1 Protocol Operations

As described in Subsection 2.1.3, SDP-VS proceeds in discrete time slots. At the end of the k-th time slot, the egress node measures average E2E delay $d_a(k)$ of this time slot. If $d_a(k)$ is greater than required delay bound T_r, the egress node enters the active mode and sends a CONGESTION_NOTIFICATION (CN) message to the ingress node which traverses the VoD streaming slice. A node in the VoD streaming slice transfers to the active mode as soon as it receives a CN message. When the CN message arrives at an in-network switch or an NFV node, the node attaches its current available capacity information to the message. The available capacity measurement module at the ingress node uses the available capacity information of all the nodes in the slice to determine the E2E available capacity. Once the ingress node receives the CN message, it sets the action of both selective caching functionality and enhanced transmission functionality as 0 for the $(k+1)$-th time slot, i.e., $a(k+1) = (0, 0)$. The purpose of caching all the enhancement-layer packets in the $(k+1)$-th time slot is to reduce the queueing delay of the video packets as soon as possible.

If the average E2E delay of the $(k+1)$-th time slot is still greater than the delay bound, the ingress node keeps on caching all the enhancement-layer packets in the following time slots until the first slot whose average E2E delay is less than T_r. Suppose the average E2E delay of the j-th time slot satisfies the delay requirement. The machine learning module is required to determine the action tuples of the following time slots. The decision is made based on the predicted traffic load and the E2E available capacity of the VoD streaming slice. If a traffic change point of cross-traffic flows is detected at a node in the active mode, the node sends an AVAILABLE_CAPACITY (AC) message with its current available capacity to the ingress node. The E2E available capacity measurement module updates the E2E available capacity when an AC message is arrived at the ingress node. Except for measuring the average E2E delay, the egress node also measures the feedback reward of executing a control action in the time slot. At the end of a time slot, the egress node sends an REWARD message to the ingress node which contains the measured feedback reward. This information is necessary to update the action-selection strategy of the machine learning module. When the congestion event is over, the ingress node sends the cached packets to the corresponding video clients by enhanced transmission. Suppose the caching buffer at the ingress node becomes empty in the h-th time slot and $a_1(h)$ is N_e (i.e., no video packet needs to be cached in the h-th time slot). The ingress node enters the deactivated mode and sends a CONTROL_DEACTIVATION (CD) message to the downstream nodes in the VoD streaming slice at the end of the h-th time slot. The node transfers to the deactivated mode when it receives a CD message. The in-network switches and the NFV nodes do not send AC message if they are in the deactivated mode. Also, the egress node stops to measure the feedback reward and send REWARD message until the

Algorithm 1 Protocol operation of SDP-VS

```
1: for each time slot do
2:    Egress node measures the average E2E delay.
3:    if the measured delay is greater than T_r then
4:       Egress node sends CN message to the ingress node.
5:       Ingress node sets the action of the selective caching functionality for the next time
         slot as 0.
6:       Ingress node sets the action of the enhanced transmission functionality for the
         next time slot as 0.
7:       Ingress node sends CA message to the egress node.
8:    else
9:       Video traffic prediction module predicts the video traffic load in the next time
         slot.
10:      Machine learning module determines the action tuple of the next time slot.
11:   end if
12: end for
```

next congestion event occurs in the network. When the nodes in the VoD streaming slice are in active mode, the protocol operation of SDP-VS is described in Algorithm 1. The responsibilities of the nodes in the VoD streaming slice are summarized in Table 3.1. The items followed by (all) are the responsibilities which are required throughout the network operation. Otherwise, the items are only required when the nodes are in the active mode.

Next, we describe the mechanism of managing the caching buffer at the ingress node. The caching buffer is operated in the FIFO manner, which means that packet cached first is sent by enhanced transmission functionality first. To better use the caching resources, the caching buffer drops the packets of the segments which have been played by the clients. The video clients periodically report the buffer information to the SDN controller in the core network [46]. Then, the controller forwards the information to the ingress node of VoD streaming slice. In this work, the buffer information is the segment number of the

Table 3.1: The responsibilities of the nodes in the VoD streaming slice

Node type	Responsibilities
Ingress node	- Action tuple selection - Control action execution - E2E available capacity measurement - Video traffic prediction - Sending protocol signalling, i.e., CD message
Egress node	- Average E2E delay measurement (all) - Feedback reward measurement - Sending protocol signalling, i.e., CN message and REWARD message
In-network switch/NFV node	- Traffic change point detection of cross-traffic flows (all) - Sending protocol signalling, i.e., AC message

video segment which is being played. When the caching buffer receives the message of buffer information, it removes the packets of the same client whose segment number is less than or equal to the segment number indicated in the message.

3.2 Video Traffic Prediction

The video traffic prediction module in Fig. 3.1 is used to predict the video traffic load for each time slot. Since the action of selective caching functionality is at layer-level, the traffic prediction module is required to predict the traffic load of each SVC layer in the following time slot. The maximum number of enhancement-layers, N_e, of all the video files stored at the video server is assumed to be identical. Thus, the dimension of the output of video traffic prediction module is $N_e + 1$. We express the prediction results for the k-th time slot

as

$$\hat{\boldsymbol{t}}(k) = \left[\hat{t}_0(k), \hat{t}_1(k), \hat{t}_2(k), \ldots, \hat{t}_{N_e}(k)\right] \tag{3.1}$$

where $\hat{t}_i(k)$ represents the predicted number of packet arrivals of layer i in the k-th time slot. The predicted traffic load of base-layer packets is denoted by $\hat{t}_0(k)$. Note that we only need to implement one traffic prediction algorithm at the ingress node and feed it with the information of each SVC layer to obtain layer-level traffic prediction. The autoregressive integrated moving average (ARIMA) model is adopted for video traffic prediction, which takes the traffic load of the past time slots as input and predicts the amount of packet arrivals in the next time slot [47, 48]. The adopted ARIMA model is denoted by ARIMA(p, d, q). In this section, we first discuss the rules of determining parameters p, q and d in Subsection 3.2.1. Then, the required traffic statistics for traffic load prediction are described in Subsection 3.2.2. At last, the ARIMA(p, d, q)-based traffic prediction algorithm is presented in Subsection 3.2.3.

3.2.1 Identification of Model Form

In this subsection, we describe the method of identifying the ARIMA model, i.e., selecting appropriate p, q and d. Based on the study in [8], the trend of a flow traffic load at the ingress node is periodic. Therefore, the model identification can be achieved by analyzing the traffic loads of the prior periods before the network operation of interest. Denote by $t_i(k)$ the actual number of video packets of layer i arrived at the ingress node during the k-th time slot. The traffic load is represented by time series $\{t_i(k)\}$. Let $\boldsymbol{t_i}(T)$ denote a

series of traffic loads observed in T time slots, given by

$$\boldsymbol{t_i}(T) = \Big[t_i(1), t_i(2), \ldots, t_i(T)\Big]. \tag{3.2}$$

Parameter d is the required number of differencing to transform time series $\{t_i(k)\}$ to a stationary time series. Let $\boldsymbol{\nabla^m t_i}(T)$ denote the m-th-order difference of $\boldsymbol{t_i}(T)$, which is expressed as

$$\boldsymbol{\nabla^m t_i}(T) = \Big[\nabla^m t_i(m+1), \nabla^m t_i(m+2), \ldots, \nabla^m t_i(T)\Big]. \tag{3.3}$$

The differencing process is given by

$$\begin{aligned}
\nabla^1 t_i(k) &= t_i(k) - t_i(k-1) \\
\nabla^2 t_i(k) &= \nabla^1 t_i(k) - \nabla^1 t_i(k-1) \\
&\vdots \\
\nabla^m t_i(k) &= \nabla^{m-1} t_i(k) - \nabla^{m-1} t_i(k-1).
\end{aligned} \tag{3.4}$$

Note that $\boldsymbol{\nabla^0 t_i}(T)$ is equal to $\boldsymbol{t_i}(T)$. The value of d is determined by conducting the augmented Dickey-Fuller (ADF) test for $\boldsymbol{\nabla^m t_i}(T)$ $(m = 0, 1, \ldots)$ [47, 49]. If the p-value of the test results for $\boldsymbol{\nabla^m t_i}(T)$ is less than a pre-determined threshold (e.g., 0.05), time series $\{\nabla^m t_i(k)\}$ is considered as a stationary series. Thus, parameter d is set as m. Otherwise, more differencing is required to transform $\{\nabla^m t_i(k)\}$ to a stationary time series. Given d and $\boldsymbol{t_i}(T)$, the selection of parameters p and q is based on the minimization of the corrected Akaike information criterion (AICC) statistic [47].

3.2.2 Traffic Statistics

Since the differenced time series, $\{\nabla^d t_i(k)\}$, is stationary, the mean of $\nabla^d t_i(k)$ is constant. Denote by μ_i the sample mean of $\boldsymbol{\nabla^d t_i}(T)$. We define a mean-corrected series, $\boldsymbol{x_i}(T-d)$, which is represented as

$$\boldsymbol{x_i}(T-d) = \Big[x_i(1), x_i(2), \ldots, x_i(T-d)\Big] \tag{3.5}$$

where $x_i(k)$ is equal to $\nabla^d t_i(d+k) - \mu_i$. The sample autocovariance function, $\hat{\tau}(z)$, of $\boldsymbol{x_i}(T-d)$ is given by [50]

$$\hat{\tau}(z) = \frac{1}{T-d} \sum_{j=1}^{T-d-|z|} \Big[x_i(j+|z|) - \bar{x}_i(T-d)\Big]\Big[x_i(j) - \bar{x}_i(T-d)\Big] \tag{3.6}$$

where $\bar{x}_i(T-d)$ is the sample mean of $\boldsymbol{x}_i(T-d)$.

3.2.3 Traffic Prediction via ARIMA(p, d, q) Model

The series of the observed traffic loads for the first $d+k$ time slots during the network operation of interest is expressed as

$$\boldsymbol{t_i}(d+k) = \Big[t_i(1), \ldots, t_i(d), t_i(d+1), \ldots, t_i(d+k)\Big]. \tag{3.7}$$

Let $\boldsymbol{\nabla^d t_i}(d+k)$ denote the d-th-order difference of $\boldsymbol{t_i}(d+k)$, which is represented as

$$\boldsymbol{\nabla^d t_i}(d+k) = \Big[\nabla^d t_i(d+1), \nabla^d t_i(d+2), \ldots, \nabla^d t_i(d+k)\Big]. \tag{3.8}$$

The ARIMA(p,d,q) model predicts the traffic load of layer i in the $(d+k+1)$-th time slot based on the previous observations. From [47], the predicted traffic load, $\hat{t}_i(d+k+1)$, is given by

$$\hat{t}_i(d+k+1) = \widehat{\nabla^d t_i}(d+k+1) - \sum_{j=1}^{d} \binom{d}{j} (-1)^j t_i(d+k+1-j) \tag{3.9}$$

where $\widehat{\nabla^d t_i}(d+k+1)$ is the prediction of $\nabla^d t_i(d+k+1)$ given $\boldsymbol{\nabla^d t_i}(d+k)$. Now, the problem is transferred to find $\widehat{\nabla^d t_i}(d+k+1)$. Given series $\boldsymbol{\nabla^d t_i}(d+k)$, the corresponding mean-corrected series, $\boldsymbol{x_i}(k)$, is represented by

$$\boldsymbol{x_i}(k) = \Big[x_i(1), x_i(2), \ldots, x_i(k)\Big] \tag{3.10}$$

where $x_i(n)$ $(n = 1, 2, \ldots, k)$ is equal to $\nabla^d t_i(d+n) - \mu_i$. Let $\hat{x}_i(k+1)$ denote the prediction of $x_i(k+1)$. Since μ_i is measured before the network operation of interest, the traffic prediction problem is finally transferred to find $\hat{x}_i(k+1)$ based on $\boldsymbol{x_i}(k)$. Time series $\{x_i(k)\}$ is an ARMA(p,q) process [47] and $x_i(k)$ can be expressed as

$$x_i(k) = w_i(k) + \sum_{m=1}^{p} \alpha_m x_i(k-m) + \sum_{n=1}^{q} \beta_n w_i(k-n) \tag{3.11}$$

where $w_i(k)$ is a Gaussian white noise with zero mean and variance $\sigma^2_{w_i}$ [47].

Next, we present the prediction method for time series $\{x_i(k)\}$. The recursive equation of finding the value of $\hat{x}_i(k+1)$ is given by [47, 48]

$$\hat{x}_i(k+1) = \begin{cases} \sum_{j=1}^{k} \theta_{k,j}[x_i(k+1-j) - \hat{x}_i(k+1-j)], & 1 \le k < m \\ \sum_{j=1}^{q} \theta_{k,j}[x_i(k+1-j) - \hat{x}_i(k+1-j)] + \alpha_1 x_i(k) + \cdots + \alpha_p x_i(k+1-p), & k \ge m \end{cases} \tag{3.12}$$

where m is the maximum of p and q (i.e., $m = \max(p, q)$). Note that $\hat{x}_i(1)$ equals 0. The coefficient, $\theta_{k,j}$, is calculated recursively by the following equations

$$\eta_0 = \kappa(1,1) \tag{3.13}$$

$$\theta_{k,k-n} = \frac{1}{\eta_n}\left[\kappa(k+1, n+1) - \sum_{j=0}^{n-1} \theta_{n,n-j}\theta_{k,k-j}\eta_j\right], \quad 0 \le n < k \tag{3.14}$$

$$\eta_k = \kappa(k+1, k+1) - \sum_{j=0}^{k-1} \theta_{k,k-j}^2 \eta_j \tag{3.15}$$

where $\kappa(h,g)$ is given by

$$\kappa(h,g) = \begin{cases} \frac{1}{\sigma_{w_i}^2}\hat{\tau}(h-g), & 1 \leq h,g \leq m \\ \frac{1}{\sigma_{w_i}^2}\left[\hat{\tau}(h-g) - \sum_{r=1}^{p}\alpha_r\hat{\tau}(r-|h-g|)\right], & \min(h,g) \leq m < \max(h,g) \leq 2m \\ \sum_{r=0}^{q}\beta_r\beta_{r+|h-g|}, & \min(h,g) > m \\ 0, & \text{otherwise.} \end{cases} \tag{3.16}$$

From (3.12) - (3.16), we can see that only parameters $\alpha_1,\ldots,\alpha_p,\beta_1,\ldots,\beta_q,\sigma_{w_i}$ are unknown. The maximum likelihood estimation (MLE) is adopted to estimate these parameters. Let $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}$ denote $(\hat{\alpha}_1,\ldots,\hat{\alpha}_p)$ and $(\hat{\beta}_1,\ldots,\hat{\beta}_q)$ where $\hat{\alpha}_i$ and $\hat{\beta}_j$ represent the estimate of α_i and β_j, respectively. Denote by $\hat{\sigma}_{w_i}$ the estimate of σ_{w_i}. To initialize MLE, we first need to obtain the preliminary estimation results of the parameters. Then, the estimates of the parameters are updated recursively. The Hannan-Rissanen algorithm is adopted to find the initial estimates [47]. The recursive equations of $\hat{\sigma}_{w_i}$, $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}$ are given by

$$\hat{\sigma}_{w_i}^2 = \frac{S(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}})}{n_o} \tag{3.17}$$

where n_o is the number of observations and $S(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}})$ is represented as

$$S(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}}) = \frac{\sum_{j=1}^{n_o}[x_i(j)-\hat{x}_i(j)]^2}{\eta_{j-1}}. \tag{3.18}$$

Estimates $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}$ are obtained by minimizing $l(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}})$ given by

$$l(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}) = \ln[n_o^{-1} S(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}})] + n_o^{-1} \sum_{j=1}^{n_o} \ln \eta_{j-1}. \tag{3.19}$$

3.3 Action-Selection via Multi-Armed Bandit

In this section, we present the learning-based method for determining the action tuple for each time slot. The objective of deploying the protocol functionalities (i.e., selective caching and enhanced transmission) at the ingress node of VoD streaming slice is to deliver more video packets without leading to network congestion. Therefore, we define the reward of executing action $a(k)$ in the k-th time slot as

$$R_{a(k)}(k) = \frac{g(k)}{r(k)T_s} \tag{3.20}$$

where $g(k)$ is the number of video packets left the VoD streaming slice in the k-th time slot whose E2E delay are less than required delay bound T_r. Through implementing different actions, the ingress node intends to maximize the expected overall reward.

The reward of executing an action in different network conditions may be different. Caching video packets during a congestion event can reduce the packet E2E delay which is beneficial to the reward. However, it is harmful to the reward if the ingress node activates selective caching functionality when the VoD streaming slice has enough resources to afford the video traffic. Therefore, video traffic load and E2E available capacity of the VoD streaming slice should be taken into consideration when the machine learning module

selecting the control actions for each time slot. We formulate this action-selection problem as a MAB problem. The video traffic load and E2E available capacity is called context information. The MAB problem which considers context information in the decision making is also referred to as contextual bandit problem [51]. The arm is the selected control action for each time slot.

Recall the protocol operations presented in Subsection 3.1.1, the action of the k-th time slot is directly set as $(0, 0)$ if the average E2E delay of the $(k-1)$-th time slot (i.e., $d_a(k-1)$) is greater than required delay bound T_r. Thus, the k-th time slot is not included in the learning algorithm. If $d_a(k-1)$ is less than T_r, the machine learning module determines the action tuple, $a(k)$, based on the context information, i.e., the outputs of the video traffic prediction module, $\hat{t}(k)$, and the E2E available capacity measurement module, $r(k)$. Let $\boldsymbol{x_{k,a(k)}}$ denote the context information of the k-th time slot which is represented as

$$\boldsymbol{x_{k,a(k)}} = \Big[\hat{t}_0(k), \hat{t}_1(k), \hat{t}_2(k), \ldots, \hat{t}_{N_e}(k), r(k)\Big]. \tag{3.21}$$

The reward of implementing action $a(k)$ in the k-th time slot is observed at the end of the slot. Then, the tuple, $(\boldsymbol{x_{k,a(k)}}, a(k), R_{a(k)}(k))$, is feed back to the machine learning module to improve the arm-selection strategy. Therefore, the learning algorithm for solving the MAB problem should specify the following factors:

- Arm-selection strategy for each time slot;
- The mechanism of improving the arm-selection strategy based on the feedback reward observed at the end of each time slot.

The LinUCB algorithm is proposed to solve the MAB problem with context information [51]. For the k-th time slot, the expected reward of arm a can be represented as

$$E\left[R_a(k)|\boldsymbol{x_{k,a}}\right] = \boldsymbol{x}_{\boldsymbol{k,a}}^T \theta_a^* \tag{3.22}$$

where θ_a^* is an unknown coefficient vector. Assume m contexts of arm a have been observed before the k-th time slot and the corresponding feedback rewards are included in response vector $\boldsymbol{R_a}$. Denote by $\boldsymbol{D_a}$ the matrix of observations whose rows represent m observed contexts of arm a. The dimension of $\boldsymbol{D_a}$ is $m \times d$ where d is the dimension of the context information. In our case, dimension d equals $N_e + 2$. The estimate of the coefficient vector, θ_a^*, is given by

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{a}} = \left(\boldsymbol{D}_{\boldsymbol{a}}^T \boldsymbol{D_a} + \boldsymbol{I_d}\right)^{-1} \boldsymbol{D}_{\boldsymbol{a}}^T \boldsymbol{R_a} \tag{3.23}$$

where $\boldsymbol{I_d}$ is the $d \times d$ identity matrix. It has been shown in [51] that, for any $\delta > 0$, the inequality (3.24) holds with probability at least $1 - \delta$.

$$\left| \boldsymbol{x}_{\boldsymbol{k,a}}^T \hat{\boldsymbol{\theta}}_{\boldsymbol{a}} - E\left[R_a(k)|\boldsymbol{x_{k,a}}\right] \right| \leq \xi \sqrt{\boldsymbol{x}_{\boldsymbol{k,a}}^T \left(\boldsymbol{D}_{\boldsymbol{a}}^T \boldsymbol{D_a} + \boldsymbol{I_d}\right)^{-1} \boldsymbol{x_{k,a}}}. \tag{3.24}$$

Parameter ξ is given by

$$\xi = 1 + \sqrt{\frac{\ln(2/\delta)}{2}}. \tag{3.25}$$

At the beginning of the k-th time slot, the machine learning module selects the arm which

Algorithm 2 Arm-selection strategy

Initialize $\xi \in \mathbb{R}_+$ and $d_a(0) = 0$.
for $k = 1, 2, \ldots$ **do**
  if $d_a(k-1) > T_r$ **then**
    Set the action tuple of the k-th time slot as $(0, 0)$.
  else
    Obtain the context information of each arm $a \in \mathcal{A} : \boldsymbol{x_{k,a}} \in \mathbb{R}^d$.
    for every $a \in \mathcal{A}$ **do**
      if a is new **then**
        $\boldsymbol{A_a} \leftarrow \boldsymbol{I_d}$
        $\boldsymbol{b_a} \leftarrow \boldsymbol{0_{d\times 1}}$
      end if
      $\hat{\boldsymbol{\theta}}_{\boldsymbol{a}} \leftarrow \boldsymbol{A_a^{-1}} \boldsymbol{b_a}$
      $\hat{R}_a(k) \leftarrow \boldsymbol{x_{k,a}^T} \hat{\boldsymbol{\theta}}_{\boldsymbol{a}} + \xi \sqrt{\boldsymbol{x_{k,a}^T} \boldsymbol{A_a^{-1}} \boldsymbol{x_{k,a}}}$
    end for
    Set the action tuple of the k-th time slot $a(k) = \arg\max_{a\in\mathcal{A}} \hat{R}_a(k)$.
    Observe the feedback reward $R_{a(k)}(k)$ at the end of k-th time slot.
    $\boldsymbol{A_a} \leftarrow \boldsymbol{A_a} + \boldsymbol{x_{k,a(k)}} \boldsymbol{x_{k,a(k)}^T}$
    $\boldsymbol{b_a} \leftarrow \boldsymbol{b_a} + R_{a(k)} \boldsymbol{x_{k,a(k)}}$
  end if
end for

can maximize $\hat{R}_a(k)$ in the following

$$\hat{R}_a(k) = \boldsymbol{x_{k,a}^T} \hat{\boldsymbol{\theta}}_{\boldsymbol{a}} + \xi \sqrt{\boldsymbol{x_{k,a}^T} \boldsymbol{A_a^{-1}} \boldsymbol{x_{k,a}}} \tag{3.26}$$

where

$$\boldsymbol{A_a} = \boldsymbol{D_a^T} \boldsymbol{D_a} + \boldsymbol{I_d}. \tag{3.27}$$

The summary of the arm-selection strategy is presented in Algorithm 2.

3.4 Summary

In this chapter, we introduce the proposed SDP-VS protocol which is used to achieve in-network control for video traffic flows in the 5G core network. First, the framework of SDP-VS including a detailed description of protocol operations is presented. Then, the ARIMA-based traffic prediction module is discussed. At last, a learning-based action-selection strategy is given.

Chapter 4

Performance Evaluation

In this chapter, we present the performance evaluation of the VoD streaming systems with and without the proposed SDP-VS. As introduced in Subsection 2.1.4, four QoS metrics are considered in the performance evaluation, i.e., average E2E delay, throughput, goodput ratio and resource utilization. First, a detailed description of the simulation settings is given in Section 4.1. Then, the numerical results of different QoS metrics are shown in Section 4.2.

4.1 Simulation Settings

The network topology considered in our simulation is presented in Fig. 4.1. Five video clients download video files from the same video server [42]. Video segment length Δ_s of all video files is 2 seconds [52]. Every segment is encoded into one base-layer chunk and four enhancement-layer chunks [53] and each chunk is assumed to be encapsulated into 200

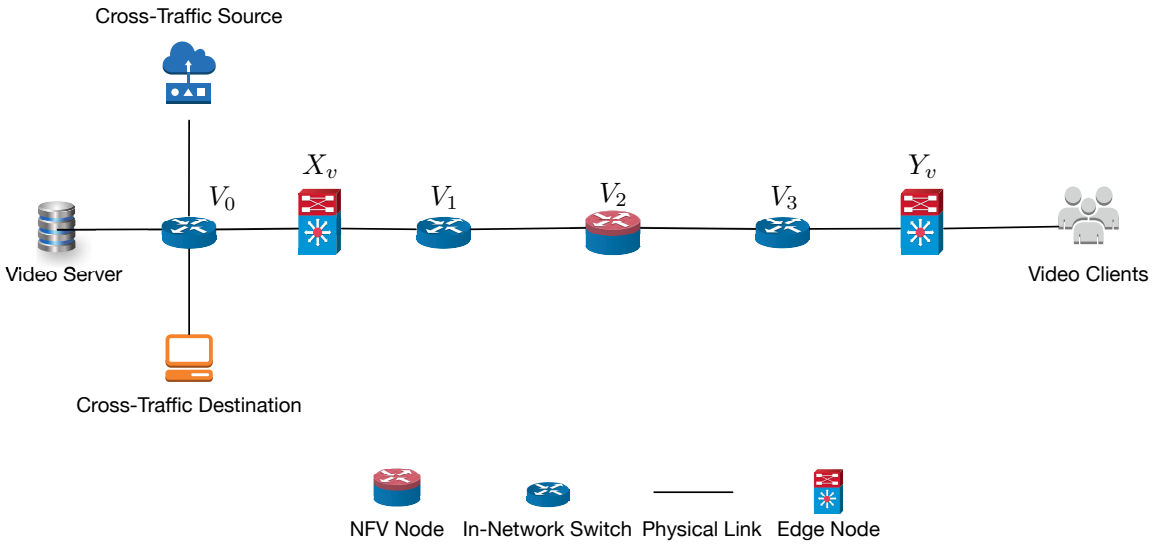

Figure 4.1: Network topology for performance evaluation.

video packets. The packet size is constant and set as 1400 bytes [42]. The aggregated video traffic flow of all five clients passes through in-network switch V_0 to ingress node X_v. Nodes X_v and Y_v are the ingress node and egress node of the VoD streaming slice, respectively. The VoD streaming slice between the edge nodes has a linear topology which contains two in-network switches (i.e., V_1 and V_3) and one NFV node (i.e., V_2). Recall the discussion in Section 2.1, the edge nodes are in-network servers which have much more resources than NFV nodes and in-network switches. Hence, nodes X_v and Y_v are not the bottleneck nodes throughout the simulation. The capacity of node V_l $(l = 0, 1, 2, 3)$ is $C_l = 4500$ packet/s [54]. The video traffic flow and the cross-traffic flow share the transmission resources at V_0. During the network operation, we change the packet arrival rate of the cross-traffic at V_0 to evaluate the performance of the proposed SDP-VS with different traffic loads. The packet inter-arrival time of the cross-traffic at V_0 in different time intervals is shown in Table 4.1.

Table 4.1: Packet inter-arrival time of the cross-traffic at V_0

Time interval	$[1, 40]$	$[41, 80]$	$[81, 120]$
Inter-arrival time	$\frac{1}{1100}$ second	$\frac{1}{1400}$ second	$\frac{1}{1700}$ second

The additive-increase multiplicative-decrease (AIMD) congestion control algorithm [55] is implemented at the video server to control the source sending rate. The propagation delay of the links outside (between) the edge nodes is set as 5 ms (2.5 ms). E2E delay bound T_r is set to 40 ms. Parameter ξ in (3.25) is 1.5 [51]. The simulation continues 120 s and the length of every time slot is 1 s. We do not consider the mechanism for caching buffer management, i.e., no cached packet is dropped during the network operation.

Before introducing the simulation results, we first identify the parameters of the ARIMA model, i.e., determining parameters p, q and d. It can be seen from Fig. 4.1 that video packets traverse the same path from the video server to the ingress node. In addition, each video chunk is encapsulated into the same number of video packets. Therefore, the ingress node only needs to predict the aggregated traffic load and divides it by the number of SVC layers (i.e., 5 in our simulation) to represent the predicted traffic load for each layer. Before the simulation, we run the network and the ingress node collects the video traffic loads of 120 time slots for data analysis. We first determine parameter d. As discussed in Section 3.2, the time series of video traffic loads becomes stationary after d times differencing. If the time series itself is stationary, parameter d is set to 0. The augmented Dickey-Fuller (ADF) test is widely used to check if a time series is stationary [47, 49]. The test results of the original time series (i.e., without differencing) are shown in Table 4.2. Since the p-value of a stationary time series should be less than a pre-determined

Table 4.2: ADF test results when $d = 0$

ADF statistic	p-value	Critical value (1%)	Critical value (5%)	Critical value (10%)
-0.913	0.784	-3.489	-2.887	-2.580

Table 4.3: ADF test results when $d = 1$

ADF statistic	p-value	Critical value (1%)	Critical value (5%)	Critical value (10%)
-9.627	1.647×10^{-16}	-3.489	-2.887	-2.580

threshold which is generally set as 0.05 [56], the time series of video traffic loads without differencing is not stationary. Then, we conduct the ADF test for the time series after the first differencing. The test results are presented in Table 4.3. It can be seen that the p-value is much less than the threshold 0.05. In addition, the ADF statistic is less than all the critical values, indicating that the time series is stationary with a 99% confidence level [56]. Thus, parameter d is set as 1 in the simulation. Then, we select parameters p and q by evaluating the AICC statistic. Based on the observed traffic loads, the AICC statistic is minimized when $p = 2$ and $q = 1$. Therefore, ARIMA$(2, 1, 1)$ model is adopted for video traffic prediction.

4.2 Numerical Results

The numerical results of average E2E delay, goodput ratio, throughput and resource utilization for the VoD streaming systems with and without SDP-VS are compared in this section. To ease representation, we denote the VoD streaming system with (without) SDP-VS by VS-W (VS-WO) system. Fig. 4.2 and Fig. 4.3 shows the average E2E delay and the goodput ratio performance, respectively. The results are obtained from ten repeated

Table 4.4: The capacity of V_2

Time interval	$[1, 20]$	$[21, 40]$	$[41, 60]$	$[61, 120]$
Congestion time = 20 s	4500 packet/s	2500 packet/s	4500 packet/s	4500 packet/s
Congestion time = 40 s	4500 packet/s	2500 packet/s	2500 packet/s	4500 packet/s

simulations. The thoughput (resource utilization) of each time slot in one simulation is presented in Fig. 4.4 (Fig. 4.5).

1. Average E2E delay: We first examine the average E2E delay of VS-W and VS-WO when a congestion event occurs in the VoD streaming slice. The network congestion is generated by reducing the capacity of V_2 from 4500 packet/s to 2500 packet/s. The capacity of V_2 in different time intervals is described in Table 4.4. Two congestion times are considered in the simulation, i.e., 20 s and 40 s. In our simulation, the capacity of V_2 is prior knowledge for the machine learning module. The simulation results are presented in Fig. 4.2. The cumulative distribution function (CDF) of the average E2E delay measured in the ten repeated simulations is used to evaluate the performance. It can be seen that the CDFs of VS-W with different congestion times are close to each other. Also, the average E2E delay of VS-W measured in all the time slots is less than the required delay bound, since the selective caching functionality is activated right after the congestion happens. By caching some enhancement-layer packets of the video traffic flow, the queue length at V_2 is well controlled. VS-W (predicted)/VS-W (real) in Fig. 4.2 indicates the simulation results of VS-W system whose machine learning module is fed with the predicted/real traffic load for each time slot. The results of VS-W (predicted) and VS-W (real) are similar to each other which can verify the efficiency of the traffic prediction algorithm. For VS-WO, the

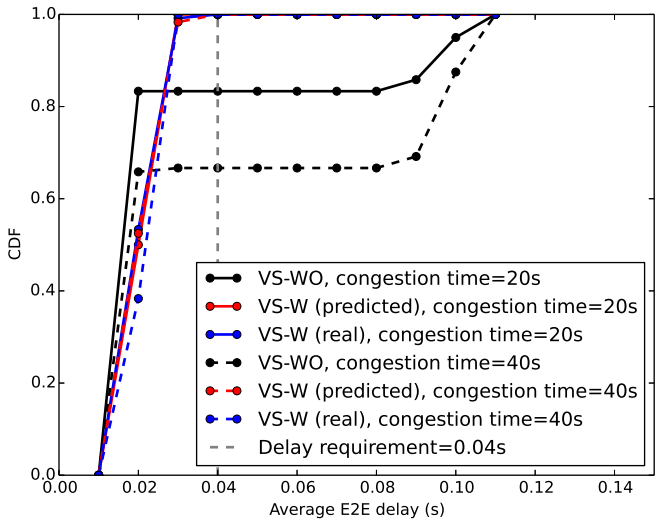

Figure 4.2: Performance of average E2E delay.

average E2E delay of around 17% time slots exceeds the required delay bound due to a 20 s congestion event. In addition, around 32% time slots do not satisfy the delay requirement when a 40 s congestion event occurs. Thus, the gap between the delay performance of VS-W and VS-WO becomes larger if the network congestion continues longer. It is observed that the CDF of VS-WO is greater than that of VS-W when the average E2E delay is 0.02 s. For VS-WO, the queueing delay is negligible after the congestion event. Hence, the average E2E delay of these time slots is in the range between 0.01 s and 0.02 s. However, the enhanced transmission functionality is activated in VS-W system after network congestion. As a result, the average E2E delay of the corresponding time slots increases to a certain extent. Note that the increased average E2E delay of the time slots with enhanced transmission does not exceed the required delay bound.

2. *Goodput ratio*: The performance of goodput ratio is also examined in our simulation.

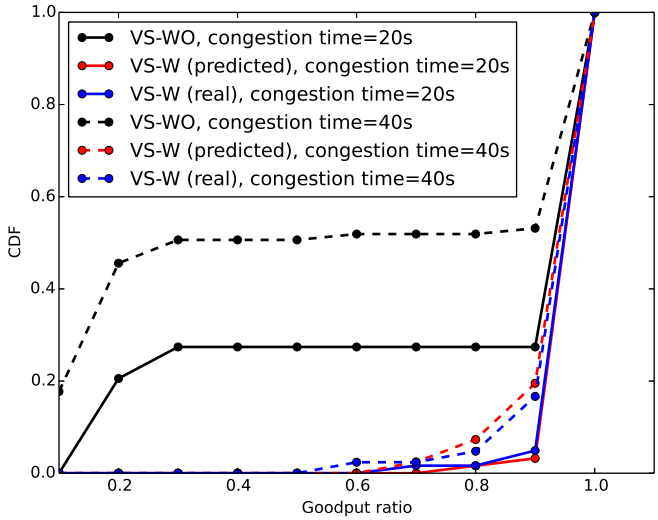

Figure 4.3: Performance of goodput ratio.

The available capacity of V_2 follows the settings in Table 4.4. It can be seen from Fig. 4.3 that VS-W outperforms VS-WO for both two congestion times. Furthermore, it is observed that the goodput ratio of VS-W is not sensitive to the congestion time, since the simulation results of VS-W for different congestion times are close to each other. The gap between the performance of VS-W and VS-WO increases with the congestion time. We also compare the performance of VS-W systems with the predicted traffic load (VS-W (predicted)) and the real traffic load (VS-W (real)). As expected, the results of VS-W (predicted) and VS-W (real) are similar to each other.

3. Throughput and resource utilization: To validate the efficiency of the proposed enhanced transmission functionality, we compare the throughput of VS-W and VS-WO for each time slot during the network operation. The machine learning module of VS-W system utilizes the predicted traffic loads in action-selection. The congestion event

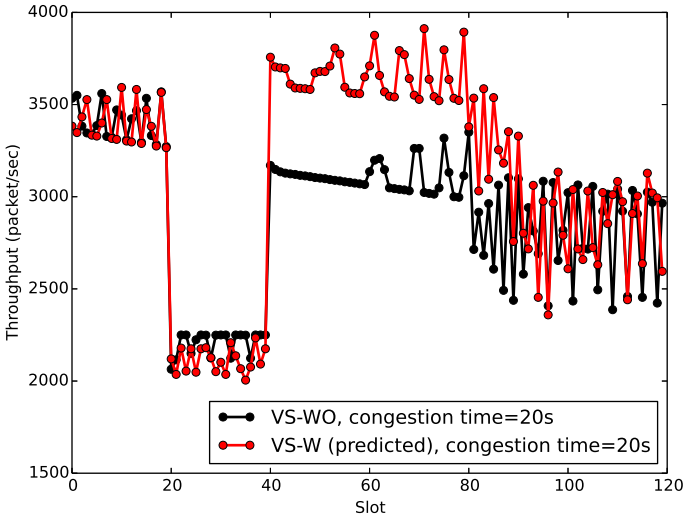

Figure 4.4: Throughput with regard to the number of slots.

exists at V_2 from 20 s to 40 s. The results are shown in Fig. 4.4. Before the congestion event, the throughput of VS-W and VS-WO is close to each other since it only depends on the video traffic load. During the network congestion, the throughput of two VoD streaming systems is also at the same level. The network congestion is mitigated after the 40-th time slot and the ingress node of VS-W starts to send cached video packets to the corresponding video clients by enhanced transmission functionality. Therefore, we can see that the throughput of VS-W is higher than that of VS-WO from the 41-th time slot. All the cached video packets are transmitted before the 91-th time slot. As expected, the throughput of VS-W returns to the same level of VS-WO from the 91-th time slot to the end of the simulation. The similar results also can be seen in Fig. 4.5 which shows the resource utilization performance of the two VoD streaming systems. The resource utilization of VS-W and VS-WO is close to each other before the 41-th time slot. Benefit

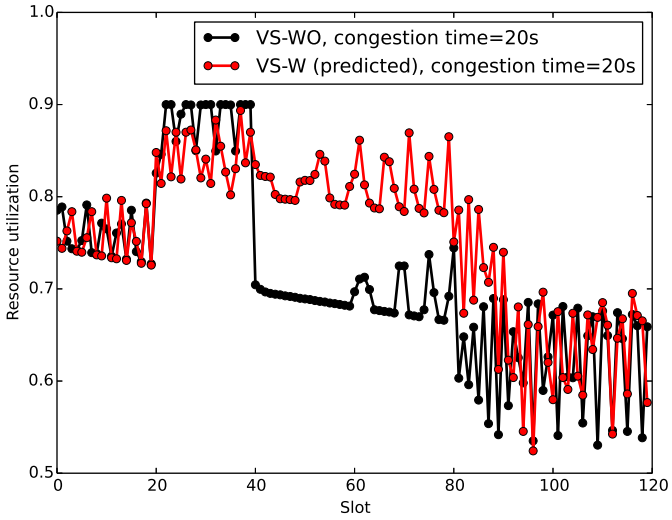

Figure 4.5: Resource utilization with regard to the number of slots.

from enhanced transmission functionality, the resource utilization of VS-W is higher than that of VS-WO from the 41-th time slot to the 90-th time slot. Then, VS-W and VS-WO have similar resource utilization until the end of the simulation.

4.3 Summary

In this chapter, we evaluate four types of performance metrics of the VoD streaming systems with and without the proposed SDP-VS. First, we examine the delay performance of VS-W and VS-WO. The results validate the efficiency of deploying SDP-VS in the core network. The advantage of VS-W becomes more prominent when the congestion event continues longer. Then, the goodput ratio of VS-W and VS-WO is compared. Results show that VS-W outperforms VS-WO for different congestion times. At last, we assess throughput

and resource utilization performance of two VoD streaming systems. The results verify the benefit of implementing the enhanced transmission functionality.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this research, we focus on designing a customized transmission protocol to control the video traffic flows in the 5G core network. The proposed SDP-VS protocol is deployed in the VoD streaming slices which incorporates three protocol functionalities, i.e., header conversion functionality, selective caching functionality and enhanced transmission functionality. To support the operation of SDP-VS, the header conversion functionality is implemented at the ingress node of VoD streaming slice which adds the SDP-VS header over all the video packets. We consider the SVC codec in the VoD streaming system that provides flexible video decoding. To better utilize the caching resources in the 5G core network, selective caching functionality puts certain video packets into the caching buffer at the ingress node in case of network congestion. Once the network congestion is over, the

ingress node sends the cached video packets to the corresponding video clients by enhanced transmission functionality. By executing the in-network protocol functionalities, the VoD streaming slice can achieve fast reaction to the network dynamics. In order to choose appropriate control actions for different network conditions, we formulate the action-selection as a MAB problem which is solved by the LinUCB algorithm. The proposed action-selection strategy takes video traffic load and available capacity of VoD streaming slice into consideration. Thus, we design an ARIMA-based traffic prediction module and an E2E available capacity measurement module to support the protocol operation. Simulation results are provided to demonstrate the advantages of the proposed SDP-VS protocol.

5.2 Future Research Work

Although this thesis proposes several protocol functionalities to control the video traffic flows in the core network, some research issues are still open:

1. In this research, we consider the scenario that the congestion event can be mitigated by caching certain enhancement-layer packets at the ingress node. If a severe network congestion occurs in the network which cannot be resolved even all enhancement-layer packets are cached, we need to find a method to reduce the source sending rate. A potential solution is to control the rate of replying the ACK packets from the ingress node to the video server. This issue is challenging because that it needs to take many factors into consideration, e.g., 1) rate control algorithm at the video server, 2) varying network conditions between the video server and the ingress node, and 3)

dynamics of the available capacity for VoD streaming slice;

2. For the current enhanced transmission functionality, a packet cached first is also sent first. However, the buffer level of video clients may be different when the enhanced transmission is activated. The client with more buffered video segments can tolerate a longer delay of receiving its cached packets. Thus, the buffer level of each video client can be considered in the future to achieve better performance in terms of user experience.

3. Similar to enhanced transmission functionality, taking the buffer level of each video client into consideration has potential benefit for selective caching functionality. To better utilize the caching resources in the 5G core network, some packets of enhancement layers from the clients with small buffer level can be dropped directly (denoted by selective dropping) when a certain congestion event occurs since these packets have small probability to be further transmitted to the clients by enhanced transmission. It is meaningful to study the relationship between buffer level and action-selection (i.e., selective caching or selective dropping). In addition, diverse video streaming applications pose different delay requirements. For the VoD streaming service considered in this research, short-term congestion events can be absorbed by the buffered video segments at the video clients [57]. Thus, VoD streaming service can tolerate certain delay when downloading the new video segment. However, some video streaming applications such as live streaming, video conferencing and video games have more stringent delay requirement. As a result, caching packets for these applications in the core network is not helpful since the cached packets have little

chance to be delivered to the clients in time by enhanced transmission functionality. Selective dropping can be a potential solution for the real-time streaming services when a congestion event occurs in the network.

References

[1] Q. Ye, W. Zhuang, X. Li, and J. Rao, "End-to-end delay modeling for embedded VNF chains in 5G core networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 692–704, 2018.

[2] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5G: RAN, core network and caching solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3098–3130, 2018.

[3] U. Fattore, F. Giust, and M. Liebsch, "5GC+: an experimental proof of a programmable mobile core for 5G," in *Proc. IEEE CAMAD*, (Barcelona, Spain), pp. 1–6, Sept. 2018.

[4] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, "Improving traffic forecasting for 5G core network scalability: A machine learning approach," *IEEE Netw.*, vol. 32, no. 6, pp. 42–49, 2018.

[5] M. R. Sama, L. M. Contreras, J. Kaippallimalil, I. Akiyoshi, H. Qian, and H. Ni, "Software-defined control of the virtualized mobile packet core," *IEEE Commun. Mag.*, vol. 53, no. 2, pp. 107–115, 2015.

[6] K. Qu, W. Zhuang, Q. Ye, X. S. Shen, X. Li, and J. Rao, "Delay-aware flow migration for embedded services in 5G core networks," in *Proc. IEEE ICC*, (Shanghai, China), pp. 1–6, May 2019.

[7] O. Alhussein, P. T. Do, Q. Ye, J. Li, W. Shi, W. Zhuang, X. Shen, X. Li, and J. Rao, "A virtual network customization framework for multicast services in NFV-enabled core networks," *IEEE J. Sel. Areas Commun.*, 2020.

[8] K. Qu, J. Chen, O. Alhussein, W. Shi, P. Yang, J. Li, S. Yan, Q. Ye, W. Zhuang, and X. S. Shen, "Learning-based software defined topology, protocol, and resource allocation for service-oriented next-generation core networks," Tech. Rep. Dec. 2019.

[9] M. Polese, F. Chiariotti, E. Bonetto, F. Rigotto, A. Zanella, and M. Zorzi, "A survey on recent advances in transport layer protocols," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3584–3608, 2019.

[10] A. Abdelsalam, M. Luglio, C. Roseti, and F. Zampognaro, "TCP Wave: A new reliable transport approach for future internet," *Computer Networks*, vol. 112, pp. 122–143, 2017.

[11] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," in *Proc. ACM SIGCOMM*, (London, United Kingdom), pp. 24–35, Aug. 1994.

[12] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth estimation for enhanced transport over wireless links," in *Proc. ACM MobiCom*, (Rome, Italy), pp. 287–297, Jul. 2001.

[13] K. Tan, J. Song, Q. Zhang, and M. Sridharan, "A compound TCP approach for high-speed and long distance networks," in *Proc. IEEE INFOCOM*, (Barcelona, Spain), pp. 1–12, Apr. 2006.

[14] S. Liu, T. Başar, and R. Srikant, "TCP-Illinois: A loss-and delay-based congestion control algorithm for high-speed networks," *Performance Evaluation*, vol. 65, no. 6-7, pp. 417–440, 2008.

[15] C. P. Fu and S. C. Liew, "TCP Veno: TCP enhancement for transmission over wireless access networks," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 2, pp. 216–228, 2003.

[16] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, *et al.*, "The QUIC transport protocol: Design and internet-scale deployment," in *Proc. ACM SIGCOMM*, (Los Angeles, USA), pp. 183–196, Aug. 2017.

[17] A. M. Kakhki, S. Jero, D. Choffnes, C. Nita-Rotaru, and A. Mislove, "Taking a long look at QUIC: an approach for rigorous evaluation of rapidly evolving transport protocols," in *Proc. ACM IMC*, (London, United Kingdom), pp. 290–303, Nov. 2017.

[18] S. Fu and M. Atiquzzaman, "SCTP: State of the art in research, products, and technical challenges," *IEEE Commun. Mag.*, vol. 42, no. 4, pp. 64–76, 2004.

[19] M. Schier and M. Welzl, "Using DCCP: Issues and improvements," in *Proc. IEEE ICNP*, (Austin, USA), pp. 1–9, Oct. 2012.

[20] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "Analytic models for the latency and steady-state throughput of TCP Tahoe, Reno, and SACK," *IEEE/ACM Trans. Netw.*, vol. 11, no. 6, pp. 959–971, 2003.

[21] N. Parvez, A. Mahanti, and C. Williamson, "An analytic throughput model for TCP NewReno," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 448–461, 2009.

[22] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (BIC) for fast long-distance networks," in *Proc. IEEE INFOCOM*, (Hong Kong, China), pp. 2514–2524, Mar. 2004.

[23] M. Šošić and V. Stojanović, "Resolving poor TCP performance on high-speed long distance links overview and comparison of BIC, CUBIC and Hybla," in *Proc. IEEE SISY*, (Subotica, Serbia), pp. 325–330, Sept. 2013.

[24] S. Ha, I. Rhee, and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 64–74, 2008.

[25] J. Gettys and K. Nichols, "Bufferbloat: Dark buffers in the internet," *Queue*, vol. 9, no. 11, pp. 40–54, 2011.

[26] K. Srijith, L. Jacob, and A. L. Ananda, "TCP Vegas-A: Solving the fairness and rerouting issues of TCP Vegas," in *Proc. IEEE IPCCC*, (Phoenix, USA), pp. 309–316, Apr. 2003.

[27] W. Zhou, W. Xing, Y. Wang, and J. Zhang, "TCP Vegas-V: Improving the performance of TCP Vegas," in *Proc. IET ACAI*, (Xiamen, China), pp. 2034–2039, Mar. 2012.

[28] Y. Guo, X. Yang, R. Wang, and J. Sun, "TCP adaptive Vegas: Improving of TCP Vegas algorithm," in *Proc. ISEEE*, (Sapporo City, Japan), pp. 126–130, Apr. 2014.

[29] J. Sing and B. Soh, "TCP New Vegas revisited," in *Proc. IEEE MICC-ICON*, (Kuala Lumpur, Malaysia), Nov. 2005.

[30] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg, "Adaptive congestion control for unpredictable cellular networks," in *Proc. ACM SIGCOMM*, (London, United Kingdom), pp. 509–522, Aug. 2015.

[31] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *Proc. USENIX NSDI*, (Lombard, USA), pp. 459–471, Apr. 2013.

[32] J. Qiao, Y. He, and X. S. Shen, "Proactive caching for mobile video streaming in millimeter wave 5G networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 10, pp. 7187–7198, 2016.

[33] T. Stockhammer, "Dynamic adaptive streaming over HTTP: standards and design principles," in *Proc. ACM MMSYS*, (San Jose, USA), pp. 133–144, Feb. 2011.

[34] K. D. Singh, Y. Hadjadj-Aoul, and G. Rubino, "Quality of experience estimation for adaptive HTTP/TCP video streaming using H. 264/AVC," in *Proc. IEEE CCNC*, (Las Vegas, USA), pp. 127–131, Jan. 2012.

[35] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5G systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 131–139, 2014.

[36] M. Moradi, W. Wu, L. E. Li, and Z. M. Mao, "SoftMoW: Recursive and reconfigurable cellular WAN architecture," in *Proc. ACM CONEXT*, (Sydney, Australia), pp. 377–390, Dec. 2014.

[37] K. Qu, W. Zhuang, Q. Ye, X. Shen, X. Li, and J. Rao, "Dynamic flow migration for embedded services in SDN/NFV-enabled 5G core networks," *IEEE Trans. Commun.*, vol. 68, no. 4, pp. 2394–2408, 2020.

[38] Z. Bozakov and M. Bredel, "Online estimation of available bandwidth and fair share using Kalman filtering," in *Proc. Int. Conf. Networking*, (Aachen, Germany), pp. 548–561, May 2009.

[39] Z.-L. Zhang, Y. Wang, D. H.-C. Du, and D. Su, "Video staging: A proxy-server-based approach to end-to-end video delivery over wide-area networks," *IEEE/ACM Trans. Netw.*, vol. 8, no. 4, pp. 429–442, 2000.

[40] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H. 264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, 2007.

[41] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proc. ACM SIGCOMM*, (Chicago, USA), pp. 187–198, Aug. 2014.

[42] S. Yan, P. Yang, Q. Ye, W. Zhuang, X. Shen, X. Li, and J. Rao, "Transmission protocol customization for network slicing: A case study of video streaming," *IEEE Veh. Technol. Mag.*, vol. 14, no. 4, pp. 20–28, 2019.

[43] J. Chen, Q. Ye, W. Quan, S. Yan, P. T. Do, W. Zhuang, X. S. Shen, X. Li, and J. Rao, "SDATP: An SDN-based adaptive transmission protocol for time-critical services," *IEEE Netw.*, 2019.

[44] N. Wu, Y. Bi, N. Michael, A. Tang, J. C. Doyle, and N. Matni, "A control-theoretic approach to in-network congestion management," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2443–2456, 2018.

[45] D. Kreutz, F. M. Ramos, P. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[46] C. W. Chen, P. Chatzimisios, T. Dagiuklas, and L. Atzori, *Multimedia quality of experience (QoE): current status and future requirements*. John Wiley & Sons, 2015.

[47] P. J. Brockwell and R. A. Davis, *Introduction to time series and forecasting*. Springer, 2016.

[48] A. Azzouni and G. Pujolle, "NeuTM: A neural network-based framework for traffic matrix prediction in SDN," in *Proc. IEEE/IFIP NOMS*, (Taipei, Taiwan), pp. 1–5, Apr. 2018.

[49] A. Pal and P. Prakash, *Practical Time Series Analysis: Master Time Series Data Processing, Visualization, and Modeling using Python*. Packt Publishing Ltd, 2017.

[50] X. Jiang and H. Adeli, "Wavelet packet-autocorrelation function method for traffic flow pattern analysis," *Comput.-Aided Civ. Infrastruct. Eng.*, vol. 19, no. 5, pp. 324–337, 2004.

[51] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. ACM WWW*, (Raleigh, USA), pp. 661–670, Apr. 2010.

[52] S. García, J. Cabrera, and N. García, "Quality-control algorithm for adaptive streaming services over wireless channels," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 1, pp. 50–59, 2014.

[53] M. Rahmati and D. Pompili, "UW-SVC: Scalable video coding transmission for in-network underwater imagery analysis," in *Proc. IEEE MASS*, (Monterey, USA), pp. 380–388, Nov. 2019.

[54] O. Alhussein and W. Zhuang, "Robust online composition, routing and NF placement for NFV-enabled services," *IEEE J. Sel. Areas Commun.*, 2020.

[55] M. Allman and V. Paxson, "RFC 5681 TCP congestion control," Tech. Rep. Dec. 2009.

[56] J. M. Weiming, *Mastering Python for Finance*. Packt Publishing Ltd, 2015.

[57] G. Tian and Y. Liu, "Towards agile and smooth video adaptation in dynamic HTTP streaming," in *Proc. ACM CoNEXT*, (Nice, France), pp. 109–120, Dec. 2012.

[58] A. K. Paul, A. Tachibana, and T. Hasegawa, "An enhanced available bandwidth estimation technique for an end-to-end network path," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 4, pp. 768–781, 2016.

[59] S. K. Khangura and S. Akın, "Measurement-based online available bandwidth estimation employing reinforcement learning," in *Proc. IEEE ITC*, (Budapest, Hungary), pp. 95–103, Aug. 2019.

[60] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: Efficient available bandwidth estimation for network paths," in *Proc. PAM*, (San Diego, USA), Apr. 2003.

[61] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[62] K. Aho, D. Derryberry, and T. Peterson, "Model selection for ecologists: the worldviews of AIC and BIC," *Ecology*, vol. 95, no. 3, pp. 631–636, 2014.