Joint Optimization of QoE and Fairness for Adaptive Video Streaming in Heterogeneous Mobile Environments

Yali Yuan[1,2] †, Weijun Wang[2,3] †, Yuhan Wang[2], Sripriya S. Adhatarao[2,5], Bangbang Ren[2,4], Kai Zheng[6], Xiaoming Fu[2*] *Fellow IEEE*

***Abstract*—The rapid growth of mobile video traffic and user demand poses a more stringent requirement for efficient bandwidth allocation in mobile networks where multiple users may share a bottleneck link. This provides content providers an opportunity to jointly optimize multiple users' experiences but users often suffer short connection durations and frequent handoffs because of their high mobility. In this paper, we propose an end-to-end scheme, VSiM, for supporting mobile video streaming applications in heterogeneous wireless networks. The key idea is allocating bottleneck bandwidth among multiple users based on their mobility profiles and Quality of Experience (QoE)-related knowledge to achieve max-min QoE fairness. Besides, the QoE of buffer-sensitive clients is further improved by the novel server push strategy based on HTTP/3 protocol without affecting the existing bandwidth allocation approach or sacrificing other clients' view quality. VSiM is lightweight and easy to deploy in the real world without touching the underlying network infrastructure. We evaluated VSiM experimentally in both simulations and a lab testbed on top of the HTTP/3 protocol. We find that the clients' QoE fairness of VSiM achieves more than 40% improvement compared with state-of-the-art solutions, *i.e.*, the viewing quality of clients in VSiM can be improved from 720p to 1080p in resolution. Meanwhile, VSiM provides about 20% improvement of average QoE.**

I. Introduction

As the prevalence of mobile devices (e.g., smartphones, tablets, and laptops) and the emerging high-rate multimedia applications including video streaming for mobile gaming [1] and social networks [2], such as Internet live broadcast and video dating, mobile video traffic increases significantly in recent years. In 2021, the total global mobile traffic is reaching 67EB per month, and it will rise to 282EB per month in 2027 [3]. Also, video traffic accounts for 69% of all mobile traffic, and it is estimated to grow to 79% by 2027 [3]. The rapid growth of mobile video traffic and user demand leads to a higher probability of multiple video streaming clients sharing a bottleneck link. The experience of multiple users can be affected greatly by the network conditions and users' high mobility, such as high fluctuation in the available bandwidth and high moving speed of clients when multiple clients simultaneously compete for the shared bottleneck link, in mobile video streaming applications [4].

This problem becomes more severe in 5G networks, where clients are subject to high mobility, the base station (BS) features typically a smaller size, and the directional antenna is often employed to prevent severe propagation loss and ensure a good transmission rate [5]. Because of the directional antenna, video content can only be transmitted when the antennas of both the base station and the mobile user are directed toward each other. In this case, handoffs[1] occur more frequently due to the small cell region (e.g., picocells with a range under 100 meters [6]) and clients' high mobility characteristics (e.g., in highway and rail environments). Frequent handoffs may cause rebuffering, which will diminish the QoE significantly. Besides, some clients may obtain lower QoE, which means there is QoE unfairness among users.

However, QoE and QoE fairness are two key metrics to evaluate the performance of video traffic for clients. QoE is an important aspect in keeping a single customer's satisfaction in isolation while QoE fairness is especially important for video content providers where operators want to keep all users sufficiently satisfied (i.e., high QoE) in a fair manner. Therefore, optimization models are needed to achieve the maximum (or at least ensure an acceptable) QoE, while ensuring fairness among users for mobile video streaming applications in terms of resource (e.g., bandwidth) allocation in shared bandwidth links.

VSiM. Based on these observations, we design VSiM, an easy-deployment and high-compatibility end-to-end solution to the QoE fairness problem in mobile video streaming applications with a shared bottleneck bandwidth. In particular, we leverage the advantages of Dynamic Adaptive Streaming over HTTP (DASH) protocol for VSiM; VSiM deployed on the server achieves QoE fairness by allocating bandwidth based

This project has received funding from the European Union's Horizon 2020 research anoinnovation programme under the Marie Sktodowska-Curie grant agreement No 824019, Horizon Europe COVER project under HORIZON-MSCA-2021-SE-01 scheme with grant agreement No 101086228, and National Natural Science Foundation of China /NSF China with grant agreement No 62172093. Most of the work has been conducted when the first co-authors were affiliated with the University of Göttingen.

Yali Yuan† and Weijun Wang† have equal contributions in this work.
Corresponding author: Xiaoming Fu* (E-mail: fu@cs.uni-goettingen.de).
[1] School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China.
[2] Institute of Computer Science, University of Göttingen, Germany.
[3] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China.
[4] College of systems engineering, National University of Defense Technology, Changsha, China.
[5] Huawei Munich Research Center, Germany.
[6] Huawei Technologies Co., China.

[1]A handoff occurs when the mobile device moves between two BSes or cells.

on the advantages of the HTTP/3 protocol. To achieve this goal, we face several challenges:

Challenge 1: How to profile the mobility impact and use the profile to maximize clients' QoE fairness in a mobile network? Most of existing works [5], [7]–[13] depend on off-the-shelf mechanisms to ensure the network performance, like QoE, by dividing bandwidth evenly among multiple clients' connections, which neglects the knowledge from clients. Clients with the same bandwidth may experience different viewing experiences in mobile video streaming applications because of clients' mobility profiles, e.g., speed, direction, and acceleration. For instance, fast-moving clients may suffer more frequent handoffs [14], [15], which causes rebuffering and reduces clients' QoE.

Mobility-profiled QoE-driven bandwidth allocation. To meet the first challenge, we leverage clients' mobility profiles and QoE-related information to design an end-to-end scheme. At the client end, each client first collects its state information including mobile profiles (e.g., speed, location, and direction) from mobile devices by GPS and Inertial Measurement Unit (IMU) [16] as well as QoE-related information (e.g., rebuffering and bitrate) from DASH video player. The collected state information is then grouped, encrypted, and sent along with the `HTTP Request` for downloading the chunk at a specific bitrate to the server. Utilizing these values and clients' QoE-related information, the proposed bandwidth allocation technique (see § III-B) chooses the optimal allocated bandwidth for each client to maximize clients' QoE fairness dynamically in a mobile environment for the real-time video streaming applications.

Challenge 2: How to satisfy the buffer requirement of buffer-sensitive clients due to their mobility? Because of the movement, after a while, some clients may be more sensitive to the playback buffer size [17]. Besides, clients may not receive the complete chunk with the requested bitrate due to the short staytime in one BS.

High-compatibility server push strategy. To tackle this challenge, we propose a novel server push approach named Slow Degrade Fast Recovery (SDFR) (see § III-C). Different from the traditional server push methods [18], [19], SDFR adds the buffer for needed clients in time dynamically without affecting the existing bandwidth allocation strategy and other clients' view quality. It is designed with a transparent mechanism that is compatible with all existing ABR algorithms. Specifically, based on clients' current staytime, handover time, and remaining buffer size, the server identifies clients who suffer high-frequent rebuffering and activates the server push function for them.

Challenge 3: How to ensure our system's robustness to support the heterogeneous mobile wireless network environment? Mobile wireless network environment is heterogeneous due to the varied topologies and varied number of BSes, as well as the varied clients' mobility patterns. The existing bandwidth allocation approaches about QoE improvement for video streaming applications [5], [9]–[13] did not consider the robustness of the model. However, It is important and useful to build a model which can be adapted to various scenarios in the real world.

Online adaptive parameter update. To overcome this challenge, we map clients' mobile profiles to hyper-parameters, such as staytime and handover time, which adapt to heterogeneous mobile wireless networks. Specifically, at the server end, the server calculates the trajectory of each client and further estimates the handover latency, staytime, and possible connection-less zones using its mobility profile and BSes' information. Furthermore, we propose the parameter update model (see § III-D), for example, based on Neural Networks (NNs), to decide the optimal parameters of the proposed model for each specific topology and the update period of bandwidth allocation.

Contribution and road map:

- We propose an adaptive end-to-end QoE fairness scheme (§ III) for the mobile video traffic with multiple mobile clients over a shared bottleneck link, named VSiM, which consists of three key techniques: 1) dynamic and fair bandwidth allocation by incorporating clients' mobile profile and QoE-related information (§ III-B); 2) quick buffer filling for clients with lower playback time according to the requirement of the buffer-sensitive clients (§ III-C); 3) adaptiveness to heterogeneous wireless network environments, like varied mobility patterns and topologies of BSes (§ III-D).
- We implement the VSiM in both simulation and prototype. The experiment results show that VSiM improves more than 40% on QoE fairness (equal to resolution improvement of clients' viewing quality from 720p to 1080p) and $\sim$20% on average of the averaged QoE compared to state-of-the-art solutions.

II. Background and Motivation

We start with the background of video streaming, including HTTP adaptive streaming, HTTP3 and QUIC protocol, Quality of Experience, and QoE fairness. We then use empirical measurements to elucidate the limitations of prior solutions and our motivations.

A. Background

HTTP/3 (HTTP/2 over QUIC). HTTP adaptive streaming (HAS) has appeared in the form of a notable standard to deliver video content over the network in the past few years [20]. HTTP/3 (HTTP/2 over QUIC) resolves the major issue of Head of Line (HoL) blocking along with multiple other improvements compared to HTTP/2. The video streaming approaches over HTTP/3 are promising, since with Quick UDP Internet Connections (QUIC), HTTP/3 has some good features like HoL Elimination, Forward Error Correction, Connection Identifier, and Server Push, benefiting today's network communications significantly. Specifically, with HTTP/3, the video streaming approaches can achieve better transmission speed, shorter loading times, and more stable connections. Especially in a mobile scene, users can enjoy more comfortable surfing on a stable and fast connection.

Quality of Experience (QoE). During video transmission, a video V is divided into a stream of smaller segments or

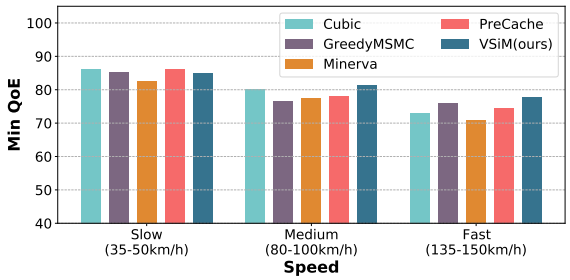


Figure 1: *High-speed clients may experience lower QoE.*

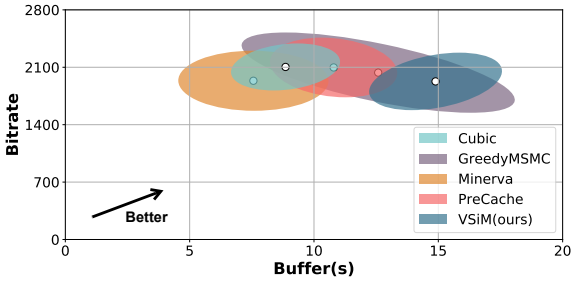


Figure 2: *The trade-off space between bitrate and buffer with the bottleneck bandwidth in a highly mobile environment.*

chunks, $V = \{1, 2, ..., K\}$ where each chunk contains S seconds of the original video. Each chunk is further encoded at different bitrates for streaming by the publisher. During streaming, the video player selects the optimal bitrate for improving the perceived video quality of the client. Higher bitrates indicate higher video quality. Hence, a common goal of video players is to request higher-quality chunks whenever the network conditions are favorable. However, the QoE of video during streaming is also affected by additional factors, especially, rebuffering and smoothness. During video streaming, rebuffering is said to occur when the video player's buffer runs out before the next chunk is downloaded, i.e., when the download time of a chunk is greater than the video player buffer's playout time. Smoothness on the other hand refers to the perceived quality variations between video segments during playtime. Hence, when requesting a video segment at higher/lower bitrates, the video players requested quality should not vary significantly from the previous one.

For a video V of length L, let c_k represents the k-th chunk at a bitrate r where $r \in \{r_1, r_2, ...r_m\}$, and R_k denote the time spent for rebuffering. Then according to the video streaming literature [21], [22], the QoE observed by a client for the k-th chunk is calculated as follows:

$$QoE\,(c_k) = q\,(c_k) - \beta R_k - \gamma ||q\,(c_k) - q\,(c_{k-1})\,||, \qquad (1)$$

where $q(c_k)$ refers to the perceived quality of the requested bitrate for the chunk c_k, $R_k = \frac{c_k}{r} - b$ refers to the rebuffering time calculated by the difference between downloading time $\frac{c_k}{r}$ and remaining playing time b. $q(c_k) - q(c_{k-1})$ represents the smoothness between the chunk c_k and c_{k-1}. The parameter β penalizes the gain in QoE with $q(c_k)$ for rebuffering while γ penalizes the QoE gain with the loss of smoothness between c_k and c_{k-1}. Therefore, as per Eq. (1), to maintain a good QoE, a video player must ensure higher bitrates, low rebuffering, and higher smoothness during video streaming.

QoE Fairness. Let B denote the bottleneck bandwidth and N be the client's number. At a period T, each client i watches a video consisting of M chunks. The total QoE of i-th client for viewing this video is denoted as QoE $_i$. Then, the max-min QoE fairness, [2] a standard QoE fairness metric, is to *maximize* $min_{i\in[N]} \frac{QoE_i}{M}$, where $[N]$ is the set of positive integers $\leq N$. Max-min QoE fairness reflects the QoE improvement of the worst performing clients, which helps service providers to offer a fairer service for clients and encourages their engagements [22]. To achieve that in a high-mobility environment, the mobility profile of each client i should take into account the resource (e.g., bandwidth) allocation.

[2]VSiM is flexible, and different QoE fairness metrics can be used to evaluate VSiM. This paper uses the max-min QoE fairness as an example to demonstrate the performance of VSiM.

B. Motivation

For mobile video streaming applications, current models, like [5], [8], [9], [11], [13], [23], with the connection-level fairness, i.e., occupying equal shared bandwidth of competing flows, may not ensure the QoE fairness for all clients, especially for those content providers with a larger number of users. In fact, in order to encourage more users to participate, video service providers are more inclined to improve the viewing quality of users with lower bitrates, rather than improving the viewing quality of users with higher bitrates [22]. Netflix, one of the largest video content providers, already considered this problem and adopts a series of techniques [24], [25] (e.g., three parallel TCP connections) to allocate a larger bandwidth for Netflix videos instead of considering connection-level fairness, reducing the rebuffering probability for low-buffer clients at video startup. Nevertheless, the fair clients' view quality among competitive network traffic is not incorporated, especially in a mobile network environment. Specifically, from the perspective of the service provider, there are two major drawbacks.

Clients' QoE can be affected by their mobility. Clearly, users may have different bandwidth allocation requirements in different scenarios [22], [26]. In mobile wireless networks, the mobility of users significantly affects network performance including the QoE and QoE fairness [27]–[32]. For instance, compared to low-speed clients, high-speed clients may require more allocated bandwidth within the same time period to accomplish the same viewing quality, due to the frequent handoffs or possible connection loss between BSes. Fig. 1 shows the view quality of clients over various speeds at time T with uniform linear motion. For simulation settings details, please refer to Section V. It is obvious to see that a client with high speed $v \in [135km/h, 150km/h]$ is more likely subject to low QoE. Specifically, compared to clients with $v \in [35km/h, 50km/h]$, the minimum QoE achieved by the clients with speeds $v \in [135km/h, 150km/h]$ is 77.64% (about 10 points with QoE normalization, see § IV-A) lower on average. Furthermore, we observe that VSiM outperforms other state-of-the-art approaches for high-speed clients by

sacrificing slightly the benefit of some low-speed clients to improve the clients with low QoE.

Mobile clients have different buffer-sensitive levels. The existing approaches with equal shared bandwidth between connections did not consider the state of the buffer-sensitive mobile video clients, like the playback buffer size. Hence, they are blind to the guidance information at the application-level, such as increasing the playback buffer size for buffer-sensitive clients. For example, a mobile video client with a short staytime in a BS will experience a handoff time or a connection loss area to go next BSes or networks. This may result in a higher chance for this client to suffer the rebuffering, which reduces its QoE in a high mobile scenario. For such kinds of clients, increasing the playback buffer size is more critical to improving their QoE compared to requesting high-quality chunks. Besides, the whole QoE fairness can also be improved because of the improvement of minimum QoE. Therefore, *dynamic* and *adaptive* buffer update strategy is a good choice to help the buffer-sensitive clients quickly replenish their buffer size.

Server push in QUIC is a promising strategy to accomplish this requirement. However, traditional server push approaches did not consider mobile characteristics in their design. Some server push strategies, like [18], [33] transmit the same quality chunk as that of the previous chunk, resulting in high downloading time, which is not suitable for mobile video clients. Besides, [33] may drop all pushed chunks and waste bandwidth resources. A novel server push strategy in VSiM is proposed to update the buffer size adaptively without affecting the existing bandwidth allocation strategy and other clients' viewing experience. Fig. 2 illustrates the trade-off space between the bitrate and buffer when facing bottleneck bandwidth in mobile environments. The larger the ellipse is, the greater the variance, resulting in the worse performance of the model. We can see that VSiM can increase the buffer size of clients significantly while slightly affecting the average bitrate.

III. VSiM System

This section introduces the design goal and overview of our system, VSiM, then discuss its three key techniques[3], namely bandwidth allocation, server push, and parameter update.

A. Overview of VSiM

VSiM is an end-to-end solution for improving the QoE and QoE fairness of video streaming in a highly mobile environment. The main design goals that we wish to achieve in VSiM are: 1) efficiently incorporate various factors in mobile environments that can potentially impact the QoE fairness of clients during video streaming, 2) easy to deploy and configure in the real world, 3) maximize the QoE fairness while ensuring the total QoE, 4) adapt to the uncertainties in heterogeneous mobile wireless networks.

System architecture. At the client end, we exploit the ABR controller ❶ to collect the bitrate of the requested chunk and the buffer state of the Dash player. We further collect information about each client's mobility profile from the sensors ❷ in their mobile devices. Since many smart devices collect such information using GPS and IMU [16], we assume that our system can also have access to such information on the clients' devices. The collected state information ❸ is then grouped, encrypted, and sent along with `HTTP Request` ❹ for downloading the chunk c_k at bitrate b_i to server.

At the server end, for each arriving `Request` from clients, the server decrypts the state information and trajectory prediction ❺ calculates clients' trajectories using mobility profile information and topology information of base stations ❻. Once the trajectory is known, the server identifies the BSes the client connects with and the associated parameters such as the handover latency, staytime, and possible connection-less zones that will impact the QoE of the mobile client. We assume that the server is aware of the information of its needed BSes. Utilizing these values and the information from clients' DASH players (e.g., buffer level and bitrate level), utility computation module ❼ applies utility function to calculate the optimal weight w_i for each client and transfer them to the bandwidth allocation module ❽ (see § III-B).

Since the server has a global view of all clients, it efficiently calculates and allocates the available bandwidth among the participating clients by considering their mobility profiles and QoE-related information. We allocate the bandwidth by the weight w_i for each client using the Cubic congestion control approach [22], [34] such that the link capacity is utilized completely and there is an improvement in the QoE fairness of all participated clients. The weight w_i is updated over a period t when the topology of BSes has a significant change. Besides, the optimal value of t and utility function parameters to quantify each factor's (e.g., bitrate, rebuffer, and smoothness) contribution for bandwidth allocation, like β and λ, are produced by the parameter update strategy ❾ (see § III-D).

Meanwhile, based on these values and information, the server also identifies the clients who are more likely to experience increased rebuffering due to a short staytime in a BS, handover latency, or connection-less zones. In order to improve the QoE of such clients, our system tries to fill their player buffer to increase playtime and thereby reduce the effect of rebuffering. Server push module ❿ identifies these potential clients, prioritizes such clients, and pushes extra chunks to them. The original chunks and the pushing chunks will be stored in two buffers (Please refer to § III-D for details).

Once the optimal bandwidth is allocated for each client and the server push gives an optimal push strategy, the prepared chunks are transmitted to the client over the allocated bandwidth using the QUIC transport protocol. However, the push bandwidth is allocated only when necessary to improve the QoE of buffer-sensitive clients for ensuring the QoE fairness of all clients and improving the efficiency of VSiM.

Discussion. VSiM may not easy to be deployed for OTT (Over-The-Top) service providers. VSiM currently can only be deployed in some applications, like military networks, autonomous mobile robot networks, and unmanned aerial vehicle

[3]The first two techniques are published in our conference paper in INFOCOM 2022

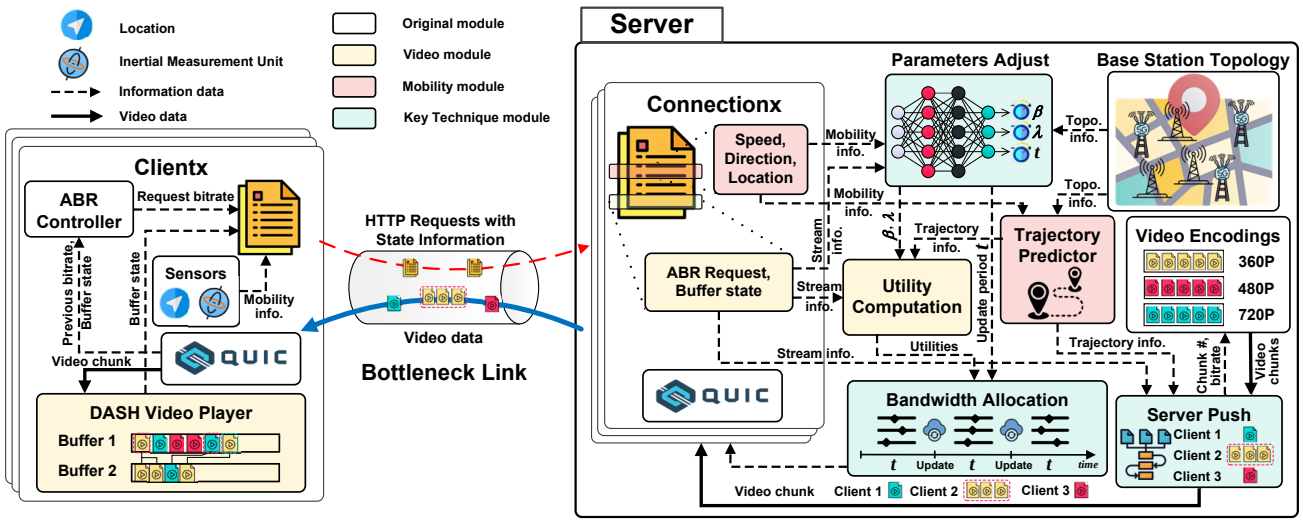


Figure 3: *VSiM improves the QoE and QoE fairness in a mobile environment by three key techniques including Bandwidth Allocation strategy considering clients' mobile profile and QoE-related information, Server Push algorithm to avoid rebuffering, and Parameter Update mechanism to adapt in heterogeneous mobile wireless networks.*

networks. In such applications, the server equipped with high storage and computation ability can be employed to optimize the overall utility of the system with the needed information, like the BSes' map and users' mobility profile. Hence, we place the mobility predictor on the server side. However, when clients and BSes share their mobility and topology information respectively to the server, privacy leakage might happen. Privacy algorithms like differential privacy can protect users' information while ensuring models' performance, but it is beyond the scope of this paper. We will consider this in our future work.

B. Bandwidth Allocation

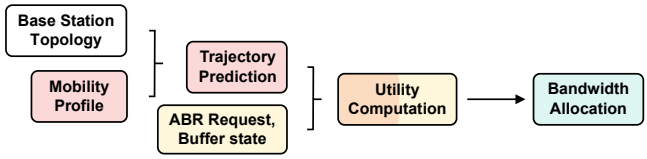


Figure 4: *Bandwidth Allocation strategy considering clients' mobile profile and QoE-related information.*

VSiM ensures a fair QoE experience for all participating clients by considering users' mobility profile, buffer level, bitrate, and smoothness during video streaming, which is described in Fig. 4. Let c_k be the current video chunk requested at a bitrate r_m and b_k be the remaining playtime in the video player's buffer, where k is the kth video chunk and m is the mth bitrate level. During video streaming, for every video segment c_k, the following state information S_{c_k} given in Eq. (2) is collected at the client module in the system consisting of QoE-related information $\{c_k, r_m, b_k\}$ and mobile profile $\{v, a, \vec{d}, l_{x,y}\}$, where c_k denotes the requested chunk, r_m is the bitrate of c_k, b_k is the buffer state, $I_{x,y}$ denotes the location. v, α, and $\vec{d}$ represents speed, acceleration, and direction.

$$S_{c_k} = \left[c_k, r_m, b_k, v, a, \vec{d}, l_{x,y}\right]. \tag{2}$$

Utility Computation. In this section, we build the mathematical model between the original QoE fairness optimization problem of VSiM and bandwidth allocation. Let $U(r)$ be the utility function for bandwidth allocation. Based on the QoE definition in Eq. (1), $U(r)$ is built as a function of each client's download rate r, which is optimized by leveraging the clients' mobility profile information and QoE-related information (e.g., bitrate level and buffer level). $U(r)$ is defined as Eq. (3).

$$U(r) = q(c_k) - \beta R'_k - \lambda |q(c_k) - q(c_{k-1})|, \tag{3}$$

where $q(c_k)$ represents the requested bitrate for chunk c_k, $\beta R'_t$ represents the rebuffering penalty, and $\lambda |q(c_k) - q(c_{k-1})|$ represents the penalty for variance in smoothness. The bandwidth allocation is dynamically changing.

Intuitively, rebuffering time aggrades along with handover time and chunk download time, and degrades along with playtime remaining and staytime. Formally, rebuffering time is:

$$R'_k = \frac{c_k}{r} - b - t_s + t_h, \tag{3a}$$

where R'_k is the total time duration. $\frac{c_k}{r}$ denotes the time to download the remaining chunk of c_k at a download rate of r, and b denotes the playtime remaining in the clients' video player buffer. The parameter $t_s = \frac{d_s}{v}$ denotes the remaining time within the connection zone of BSes, where v is the client's moving speed and d_s is the remaining distance within the connection zone of BSes. t_h denotes the handover latency between the current and the next BS. Therefore, We mapped users' mobility characteristics to the rebuffer parameter R'_k. Both the staytime t_s and handover time t_h are decided by users' mobility profile, like the speed, direction, and acceleration. The higher t_s is, the higher QoE while t_h is inversely proportional to QoE.

The handover time t_h is defined as:

$$t_h = \begin{cases} \frac{d_h}{v} + \tau, & \text{no overlap between BSes,} \\ \tau, & \text{overlap between BSes,} \end{cases} \tag{3b}$$

where d_h is the distance occurring connection loss between BSes. τ is the time when clients switch between BSes. For example, a client travels from its current BS to the next BS, if there exists the connection loss between these two BSes, then $t_h = \frac{d_h}{v} + \tau$. Otherwise, $t_h = \tau$. Please note that since the trajectory of each client is varying over time by changing the speed, direction, and acceleration, the calculation results of both t_h and t_s are also varying over time.

Bandwidth allocation. The information required for Bandwidth allocation is described in Fig. 4. Besides, given the above definition for the utility computation, the bandwidth weight for client i is calculated as $w_i = \frac{r_i}{\widetilde{U}(r_i)}$ where small$\widetilde{U}(r_i) = \frac{U(r_i)}{U(B)}$ and the allocated bandwidth will be $r_i = \frac{w_i}{\sum_{i=1}^{n} w_i} B$, where i, n, and B represents the i-th client, clients number, and server's total bandwidth. We put the convergence proof of bandwidth allocation in the Appendix (§ VIII). Please note that the time complexity of VSiM is very small, *i.e.*, $O(cn)$, where n is the number of clients sharing the bottleneck link and c is the iteration times required to converge to the optimal bandwidth allocation that maximizes the QoE fairness. The space complexity is also very small since clients' state information is refreshed on the server for each bandwidth allocation.

C. Server Push

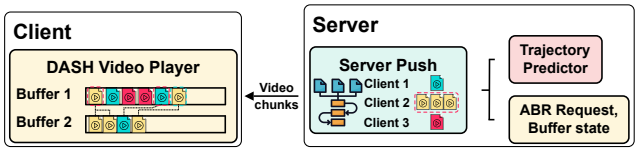


Figure 5: *Server push module utilizes clients' mobility and video stream info to determine if and how to push extra chunks.*

The server push module is employed to decrease the frequency of rebuffering for buffer-sensitive mobile video clients. It significantly increases these clients' playback buffer by pushing multiple lower-bitrate chunks when they drop into an emergency situation. Meanwhile, server push should be compatible with arbitrary ABR algorithm and should not offset the benefit from bandwidth allocation. Therefore, we carefully design a novel server push algorithm called Slow Degrade Fast Recovery (SDFR). The core thinking includes 1. *Multiple chunks encapsulations.* The server encapsulates multiple lower-bitrate chunks back to the client according to the client's state. 2. *Slow degrade.* During server push, the bitrate of the pushing chunk degrades level by level to control the smoothness.

Workflow. Fig. 5 illustrates the overview of server push. At the clients' end, the Dash player maintains two buffers logically, in which Buffer 1 stores the original request video chunks and Buffer 2 stores the pushing video chunks, recording the corresponding relation of each chunk in Buffer 2, i.e., the encapsulation relation, and delivers fake bitrate information to ABR algorithms. At the server end, after receiving `HTTP Request` message from the client, the server first estimates

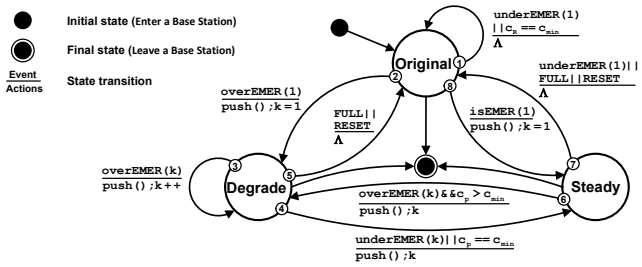


Figure 6: *Illustration on the server push algorithm Slow Degrade Fast Recovery (SDFR) as a state machine.*

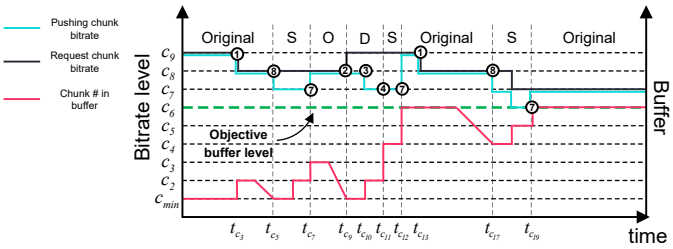


Figure 7: *Illustration on an example of one client's server push process (the dotted lines split its states).*

the client's state with the buffer state information and the trajectory predicted by the mobility profile from the message. Then, it requests and encapsulates multiple lower-bitrate chunks from video encoding (i.e., video dataset) according to the SDFR algorithm and the bitrate request from `HTTP Request` message. Fig. 5 depicts a toy example: Client 2 requests one 720P bitrate chunk while server push encapsulates and responds to three 360P chunks; the Dash player stores three 360P chunks into buffers and sends a piece of fake information "received one 720P bitrate chunk as request" to ABR algorithm.

Fig. 6 shows the server push algorithm with a state machine model and Fig. 7 illustrates an example of a client's state transitions. SDFR sorts the bitrate level with a descending order resulting in a list $[c_{max}, c_2, \cdots, c_{min}]$. The bandwidth demand $B(\cdot)$ of these bitrate levels follows a total order: $B(c_i) < B(c_j)$ if $c_i < c_j$. c_p and c_R denote the pushing chunk bitrate and the client's bitrate request, respectively. Variable `k` quantifies client's emergency level. `isEMER(k)` denotes the event of the k-level emergency, whose condition is $\texttt{t}_\texttt{h}\texttt{-b==k*t}_\texttt{s}$ (The physical meaning is that the client needs to download *extra* k chunks per request in the following staytime such that its playback buffer have enough video to play during the handover time). Similarly, `overEMER(k)` and `underEMER(k)` denote the event of emergency exceeds k-level or lacks k-level, whose conditions are $\texttt{t}_\texttt{h}\texttt{-b>k*t}_\texttt{s}$ and $\texttt{t}_\texttt{h}\texttt{-b<k*t}_\texttt{s}$, respectively. SDFR also receives *messages* from other modules: bitrate request of ABR algorithm c_R in `HTTP Request` from client; `FULL` when client's buffer is full; `RESET` when trajectory prediction module detects t_h changes. The server maintains a server push state machine for each client and the state transition happens when the server receives the corresponding client's `HTTP Request`. Below, we describe each state and transition.

- **Initial.** Client enters a base station, then server push starts

execution and changes to `Original`.

- **Original: responding chunk with `Request` bitrate.** Server push holds sending chunk with c_R bitrate. It stops at `isEMER(1)` (to `Steady`) or `overEMER(1)` (to `Degrade`). If t_h==0, it changes to `Final`.
- **Degrade: estimating the emergency level.** Degrading c_R directly to c_p may significantly decrease client's QoE if $c_R >> c_p$. To allow a smooth decrease, SDFR probes the emergency level by level, i.e., adding one additional chunk controlled by `k++`. A larger k allows more chunk delivery. During emergency, server push runs the `push()` procedure to encapsulate k chunks with bitrate c_p into `HTTP Response` message, where c_p satisfies $\max\ c,\ s.t.\ k \cdot B(c) \leq B(c_R)$. After emergency estimated (`underEMER(k)` || c_p==c_{min}), server push changes to `Steady`. If server push receives `FULL` message or `RESET` message, it changes to `Original`. If t_h==0, it changes to `Final`.
- **Steady: multiple chunks pushing.** Client achieves a high buffer fill rate by server push encapsulating and sending multiple c_p bitrate chunks with $B(c_R)$ bandwidth. It changes to `Degrade` when `overEMER(k)` && c_p>c_{min}. If `underEMER(1)` or server push receives `FULL` message or `RESET` message, it changes to `Original`. If t_h==0, it changes to `Final`.
- **Final.** Client leaves a base station, server push ends.

D. Parameter Update

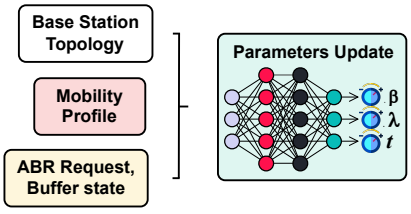


Figure 8: *Parameter update module uses BS and clients info to update VSiM's parameters by neural network algorithms.*

The parameter update module is one of the key contributions for VSiM, which is employed to generate the optimal values of parameters required by the bandwidth allocation module (§ III-B) based on the BS topology (*e.g.*, distances and number), users' profile (*e.g.*, speed, direction, and number), ABR request information (*e.g.*, bitrate), and buffer state (*e.g.*, remaining buffer size), given in Fig. 8. The updated parameters include weights β and λ of the utility function in Eq. (3) denoting the contribution of factors (*e.g.*, bitrate, rebuffer, and smoothness) to control a mobile video's rate shares. Besides, these parameters' updated frequency t is also produced by the parameter update module.

In VSiM, the parameter update module, i.e., parameters (e.g., β and λ) matching different scenarios, should be operated first once a new application scenario appears. In real-world application scenarios, β and λ might differ according to users' demand, service, content type, topology, end-service type, etc. For example, the high value of β would be better for delay-sensitive applications. Experts can set β and λ in advance considering different factors.

Many factors can impact the parameters selected for varied scenarios distributed in different dimensions. Besides, the relationship between the multi-dimensional features is complex and non-linear. Thus, deep learning approaches (*e.g.*, Neural Networks for Multi-Output Regression [35]) can be utilized in this module for predicting the suitable parameters (e.g., β and λ) for a specified application scenario.

The parameter update module has two main procedures. (1) constructing the training dataset. We take different parameter-simulating values (e.g., β and λ) for different scenarios. For example, $\beta \in \{0, 1, 2, \cdots, 50\}$ and $\lambda \in \{0, 0.1, 0.2, \cdots, 1\}$ are randomly assigned to different scenarios. In this way, we fixed the β and λ for each scenario[4]. After fixing the parameters for the scenario, the bandwidth allocation strategy can be performed. We run numerous (e.g., 3000) simulation instances to collect the training data[5] $\{\vec{v}, \vec{d}, \vec{n}, q(\vec{c_k}), q(\vec{c_{k-1}}), ...\beta, \lambda, t\}$, where $\{\beta, \lambda, t\}$ can be treated as labels. The rest can be used as features. $\{\vec{v}, \vec{d}, \vec{n}, q(\vec{c_k}), q(\vec{c_{k-1}}), ...\}$ are the clients and BS related profile information. $\vec{v}$, $\vec{n}$, and $\vec{d}$ are the speeds of clients, number of clients or BSes, and distances among BSes. (2) predicting parameters for a new scenario. We trained deep learning models (e.g., Neural Networks for Multi-Output Regression). Once a new scenario appears, we first collect the related information, like the user's profile (e.g., speed, direction, and number) and BS topology information (e.g., distances among BSes, number, etc.), and input these features to the trained models to predict a better value of parameters for such kind of scenario. After we fixed the parameters in a specific scenario, the bandwidth optimization process can be performed accordingly.

IV. Implementation

We implement VSiM in both simulation and prototype. VSiM sits between the low-level functions (QUIC protocol) and the high-level applications (Dash video player on the client end and Video encoding on the server end). On the client end, VSiM modifies Dash player (Version 3.1.0) [36] by maintaining two buffers logically [6]. On the server end, all the modules of VSiM in Fig. 3 are implemented in Go language (Version 1.13.8) based on QUIC_GO (Version 0.17.1) [37]. Two penalty parameters for rebuffering and smoothness in Eq. (3) are $\beta = 20$ and $\gamma = 0.1$ for VSiM without triggering the parameter update strategy while these two values are set as $\beta \in [0, 50]$ and $\lambda \in [0, 1]$ for VSiM over varied topologies. The period time of parameter update in the bandwidth allocation strategy (§ III-B) is $t \in [1, 15]$. These values are set according to the simulation experience. Besides, the movement of clients incorporates three groups: slow movement ($v \in [35, 50] km/h$), medium movement ($v \in [80, 100] km/h$), fast movement ($v \in$

[4]In the real world, we can get this information from experts.

[5]$\{t_s, t_h\}$ reflects mobility characteristics calculated based on users' profile and BSes Topology

[6]Note that, this modification does not break the easy deployment characteristic of VSiM because this modification is implemented on the server and the buffer allocation happened at the start of clients building connection with the server.

$[135, 150]km/h$). The accelerations of car/motorcycle and train are within $[-8, 2.5]m/s^2$ and $[-7, 0.5]m/s^2$, respectively. For the MLP model, the number of hidden layer is one with 100 neuros. The activation function is rectified linear function. The entropy loss function is employed.

A. System Settings, Metrics, Dataset, and Benchmarks

System settings. We use a server equipped with an Intel Core i7-5930K CPU at 3.5GHz, 32GB (DDR4 3000MHz) of RAM, Killer E3000 2.5Gbps Ethernet network port. All clients use Google Chrome (Version 83) with QUIC (HTTP/3) support enabled. We use 10 devices including iPhone XR, Xiaomi Mi 8, Surface Go 2, 2 $\times$ IPad Air 4, IPad Mini 4, ThinkPad X1, and 2 $\times$ MacBook Pro, as the mobile clients for the prototype test (see § V-C).

Evaluation Metrics. To better see the performance of VSiM, we regularize the scope of QoE within [0,100] [22] by Equation $a \times ln(x) - b$, where $a = 16.61$ and $b = 42.94$ for our employed datasets. For instance, about 5.8 points improvement with QoE normalization can accomplish a video quality jump from 720p to 1080p. It is defined by a large number of experimental statistics over the employed dataset. VSiM is evaluated by the following metrics: 1) QoE: It is employed to describe clients' viewing experience for the mobile streaming video, calculated in Eq. (1) (see § II-A). 2) Max-min QoE fairness: It reflects the QoE improvement of clients with the minimum QoE (see § II-A). 3) Minimum QoE: It represents the QoE of the client with the minimum value among all clients. 4) Average QoE: it is the average QoE overall participated clients. 5) Cumulative Distribution Function (CDF): It reflects the QoE fairness improvement.

Datasets. VSiM works well on videos with varied bitrates from different sources. In this paper, we evaluate VSiM by a standard test dataset [38], which reflects the real-world distribution. It includes 20 videos [7]. The value of $c_k$in our dataset ranges from 45kbps to 3936kbps. It includes low, middle, and high levels of bitrates.

Benchmarks. We have four benchmarks for comparison to prove the performance of VSiM: (1) Cubic [34] with the average bandwidth; (2) Minerva [22], where QoE fairness is targeted for video streaming with a bottleneck link but without mobility consideration; (3) GreedyMSMC [11] achieving the QoE improvement by leveraging the load balance in base stations in a mobile environment; (4) PreCache [5] improves the QoE performance by pre-storing video in next base station's cache in mobile wireless networks. Cubic, GreedyMSMC, and PreCache are originally suitable for mobile scenes. For Minerva, we transplant its utility function and perceptual quality concept in our mobile experimental scenes. All algorithms are implemented in the same mobile environment for comparison.

B. Mobility Pattern

VSiM adapts to various mobility patterns, which are mapped to staytime and handover time, to fulfill the QoE fairness for high mobile clients. In this paper, we use three mobility models as examples to evaluate our mechanism. 1) freeway mobility model [30] and railway mobility model [39]: mobile users are restricted to their lanes on the freeway/railway and its velocity is temporally dependent on its previous velocity; 2) random waypoint model [40]: at every instant, a user randomly selects a destination and moves towards it with a velocity selected uniformly randomly from $[0, v_{max}]$, where $v_{max}$is the preset maximum velocity for each user. It is commonly used in simulations.

[7]This table is in Appendix (§ VIII-D).

V. Evaluation

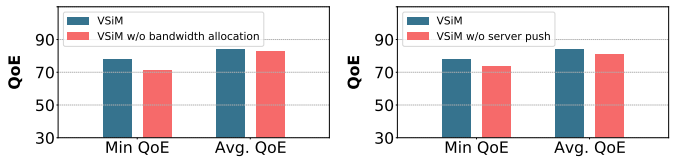
In this section, we first introduce the simulation and prototype scenario. Then, we verify the contribution of each key technique in VSiM and the robustness of VSiM by simulations. Following that, the comparison of VSiM against state-of-the-art solutions with various metrics is illustrated over a prototype wireless network. For all results, we repeated the experiment for each bandwidth for 20 runs. Please note that if there is no specified topology, topology 1 is employed to evaluate the models' performance. The parameter update strategy of VSiM is only triggered when the topology is changed.

A. Simulation and Prototype Scenarios

Simulations and prototype tests are implemented along the railway or highway direction. Two different scenarios to verify our system are as follows. For 5G BSs, we use the stand propagation and path loss model $PL_1 = 28.0 + 22log_{10}(d) + 20log_{10}(f)$ given in 3GPP TR 38.901 V16.0.0 (2019-10) [41]. The carrier frequency is 3.5 GHz, the shadow fading std in Macro BS is 4dB. For 4G BSs, we use the Marco cell propagation model [42]. The BS antenna height is fixed at 15m, and a carrier frequency of 2 GHz is used. The path loss model is $PL = 128.1 + 37.6log(d)$ with slow fading deviation in Macro BS as 10dB. Other parameters are set according to these 3GPP standards.

Topology 1: BSes with the connection loss area. We select the railway and highway from the train station in city A to the train station in city B (around $110km$). Along the road, we give the assumption that this area is covered with 237 BSes consisting of 37 4G BSes and 200 5G BSes. The reason is that the 5G BSes are deployed every 500 meters and 4G BSes every 3km, depending on the communication range of BSes and area requirements (*e.g.*, high density of BSes in an urban area while low density in a rural area). The transmission ranges of 4G and 5G BS are $2km$ and $300m$, respectively [43].

Topology 2: BSes without connection loss area. All BSes have a perfect overlap in a developed urban area. We select the railway and highway from city A to city B (around $24km$). Along the road, we give the assumption that this area is covered with 48 5G BSes, which are deployed every 500 meters depending on the communication range of BSes. The transmission range of 5G BS is $300m$



(a) Bandwidth allocation **(b) Server push**

Figure 9: *Bandwidth Allocation and Server Push techniques bring contribution to VSiM QoE improvement on both QoE fairness and average QoE in Topology 1.*

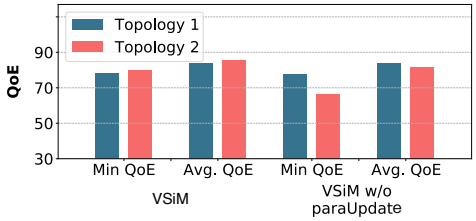


Figure 10: *Parameter update technique brings contribution to VSiM's QoE improvement on both QoE fairness and average QoE over varied topologies.*

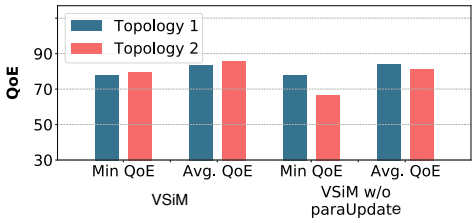
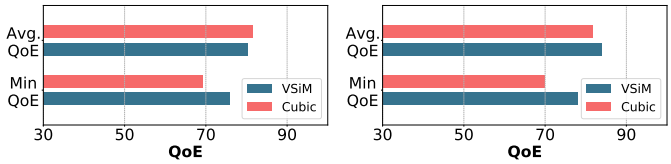


Figure 11: *Parameter update technique brings contribution to VSiM's QoE improvement on both QoE fairness and average QoE over varied topologies.*



(a) Topology 1. **(b) Topology 2**

Figure 12: *VSiM can handle various topologies and maintain a sizeable gain on QoE fairness.*

B. Baseline Comparison

We verify the contribution of each key technique in VSiM and the robustness of VSiM by simulations, where 20 to 60 clients are employed in the topology 1 and 2.

Contribution of each key technique.

We first measure the contribution of VSiM's two key techniques, i.e., Bandwidth Allocation and Server Push, with 60 clients and 150Mbps bandwidth in Topology 1. The results are described in Fig. 9. Then, the third technique, i.e., parameter update, is triggered in VSiM when the topology is changed, which is verified in Fig. 11 with 60 clients and 150Mbps bandwidth.

Specifically, in Fig. 9(a), we observe that the minimum QoE in VSiM is about 33% (about 4.8 points with the QoE normalization) higher compared to that in VSiM without the bandwidth allocation technique while accomplishing a desirable average QoE. This is equivalent to the minimum viewing quality of clients in VSiM without the bandwidth allocation strategy is 240p while the minimum viewing quality of clients with that is 360p, thanks to the bandwidth allocation technique (§ III-B), which leverages users' mobility profiles, requested bitrate, and playback buffer size to allocate the bandwidth fairly among clients. In Fig. 9(b), we notice that the minimum QoE and average QoE per client in VSiM is about 15% (about 2.4 points, equal to a viewing experience jump from the bitrate level 2944kbps to 3340kbps in resolution 1080p) and 13% higher than that in VSiM without the server push technique. Thanks to our proposed SDFR server push strategy (§ III-C) given the current buffer level, staytime, and handover time. This mechanism greatly improves the QoE fairness of clients.

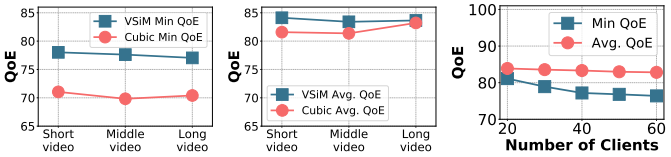
Fig. 11 gives the minimum QoE and average QoE comparison between VSiM and VSiM w/o paraUpdate over different topologies. We notice that both VSiM and VSiM w/o paraUpdate perform well in topology 1 while VSiM achieves a higher value in terms of both minimum QoE and average QoE compared with those of VSiM w/o paraUpdate in Topology 2. This is because we assign a set of selected optimal parameters (e.g., $\beta = 20$ and $\lambda = 0.1$) for both VSiM and VSiM w/o paraUpdate regarding Topology 1. However, when the topology changes, the parameter update strategy is triggered in VSiM to adapt to different topologies dynamically. As for VSiM w/o paraUpdate, the previously selected parameters for Topology 1 may not be suitable for Topology 2.

Robustness of our system. VSiM is robust over various uncertainties, like different video lengths and BSes' topologies, different ABR algorithms, and different clients' mobility patterns and the number of clients.

• *Impact of various topologies.* In Fig. 12, the minimum QoE and average QoE of VSiM and Cubic over two different topologies (§ V-A) over 60 clients and 150M bandwidth. It is obvious that compared to Cubic, the minimum QoE in both topologies A and B of VSiM achieves a significant improvement of the minimum QoE (about 6 points on average, equal to the clients' viewing experience with a resolution jump from 720p to 1080p). Besides, The average QoE of VSiM is close to that of Cubic in these two topologies.

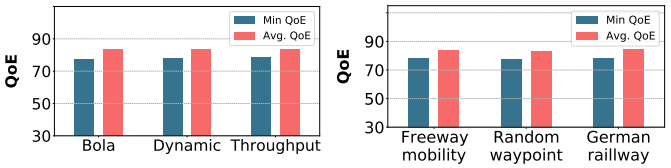
• *Impact of various video lengths.* In Fig. 13(a) and (b), we observe that VSiM accomplishes a stable minimum QoE and average QoE over various lengths of videos, which are much better compared with those of Cubic, especially for the minimum QoE.

• *Impact of large-scale clients numbers.* In Fig. 13(c), we find that the performance (*e.g.*, minimal QoE or average QoE per client) of VSiM is almost constant under various numbers of clients, showing the stability of VSiM against the variant number of clients. The slightly reducing trend of the minimum QoE and average QoE with the increasing number of clients in Fig. 13(c) is caused by the probability that the greater the number of users, the greater the probability that some clients



(a) Various video lengths (Min. QoE). **(b) Various video lengths (Avg. QoE).** **(c) Various clients scale.**

Figure 13: *VSiM achieves high QoE under various video lengths; It also can ensure stable and high QoE fairness and average QoE under a large-scale number of clients.*



(a) Various ABR algorithms. **(b) Various mobility patterns.**

Figure 14: *VSiM maintains stable and high QoE under various mobility patterns of clients and various ABR algrithms.*

will obtain lower QoE.

• *Impact of various ABR algorithms.* VSiM is transparent to ABR algorithms. In Fig. 14(a), we can clearly see that VSiM achieves good performance over different ABR algorithms regarding both the minimum QoE and average QoE.

• *Impact of various mobility patterns.* In VSiM, we convert the users' mobility profiles to the staytime and handover time, which adapts VSiM to various mobility patterns, which is given in Fig. 14(b). Besides, a new parameter adjustment model is proposed to ensure VSiM adapts to various BSes topologies and the number of nodes.

C. Prototype Test

We build a prototype test in a lab testbed to check VSiM's performance in real-world scenarios in the Topology 1 (§ V-A) over 10 clients with bandwidths [10Mbps, 15Mbps, 25Mbps, 35Mbps] bandwidth. We run the experiment under a multi-user scenario that travels between two German railway stations with a $110km$ distance and runs VSiM over an actual wireless network link in mobile networks.

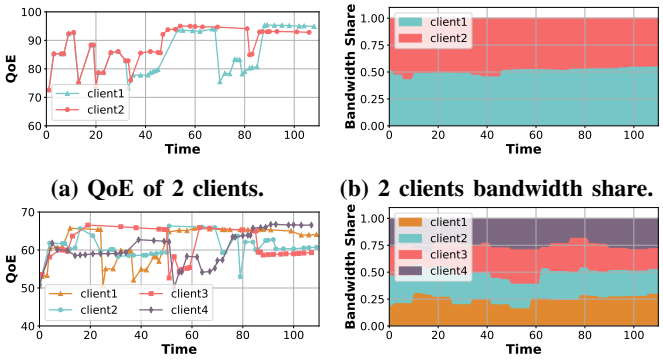
Sensitivity to network settings. The impact of network uncertainties, like bandwidth variance and latency variance, are tested in this section.

• *Impact of bandwidth variance.* We report the bandwidth and QoE variations over time by a real wireless link in mobile networks at bandwidth 10Mbps with two or four mobile clients in Fig. 15 to show how the system assigns the bandwidth and the impact of bandwidth allocation on the QoE changes.

In Fig. 15(a) and (b), we observe that both the average QoE and allocated bandwidth of two mobile clients C_1 and C_2 are close at an initial period of time t (*e.g.*, $t \in [0, 60s]$). This is because both C_1 and C_2 at this period of time are moving inside the BS with a long staytime (*e.g.*, greater than $60s$). In this case, we give the same staytime value (*e.g.*, 60s) for these two clients, which is given to avoid one client occupying the whole bandwidth and further improve QoE fairness. Then, after some time, C_2 is still inside the BS, but the staytime of C_1 is short

Algorithm	100ms	200ms	300ms
VSiM (Min QoE)	**76**	**73**	**68**
Cubic (Min QoE)	69	67	63
VSiM (Avg. QoE)	**82**	**80**	**78**
Cubic (Avg. QoE)	80	78	73

Table I: *VSiM maintains high Minimum (Min) and Avg. (Average) QoE than Cubic under various latency conditions.*



(a) QoE of 2 clients. **(b) 2 clients bandwidth share.**

(c) QoE of 4 clients. **(d) 4 clients bandwidth share.**

Figure 15: *QoE and bandwidth allocation of VSiM videos by a real wireless link over 10Mbps bandwidth in mobile networks.*

and may go to the next BSes or experience some connection loss area because of the fast movement. VSiM captures this and utilizes the optimization strategy to improve the allocated bandwidth and QoE of C_1. In Fig. 15(b), it is clear that C_1's bandwidth increases. Because of the fixed total bandwidth, C_2's bandwidth is reduced.

Similarly, in Fig. 15(c) and (d) with 4 clients, we can see that after some time, C_1 and C_4 are allocated with higher bandwidth, which improves their QoE. Because of their mobility, they may experience low viewing quality (*e.g.*, experience connection loss zone or frequent handoffs) after some time. VSiM improves the clients with lower viewing quality to maximize the QoE fairness for all clients.

• *Impact of latency variance.* Table I illustrates the impact of latency variance on VSiM and Cubic. For Table I, we observe that VSiM achieves better performance in terms of both the minimum QoE and average QoE than those of Cubic.

Compare with state-of-the-art. In this section, we compare VSiM with state-of-the-art regarding the average QoE, QoE fairness, and CDF over various bandwidths.

• *Average QoE Comparison.* In Fig. 16, we observe that: 1) increasing the bandwidth will improve the mobile client's total QoE. This is because a higher bandwidth value leads to the DASH requesting a higher bitrate, which further improves each client's QoE; 2) VSiM outperforms state-of-the-art solutions with ten mobile clients over different bandwidths regarding the average QoE.

Specifically, the average QoE improvement at bandwidth 25Mbps of VSiM is about 13% (about 2.0 points) and 30% (about 4.3 points) higher than that of Cubic [34] and Minerva [22]. In comparison, around 11% (about 1.8 points) and 16% (about 2.4 points) on average improvement is achieved by VSiM compared with PreCache [5] and GreedyMSMC [11]. VSiM with more than 2.0 points improvement can at least

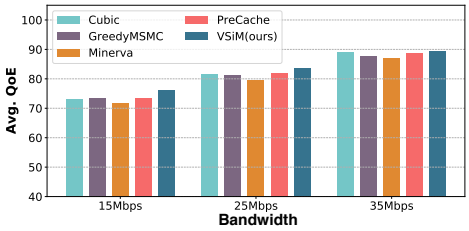


Figure 16: *VSiM fulfills a higher average QoE than various algorithms over different bandwidths.*

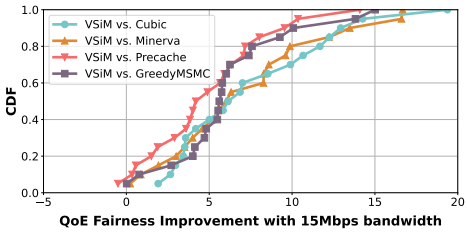


Figure 17: *QoE fairness improvement achieved by clients under various algorithms with 15Mbps bandwidth.*

jump one bitrate level compared with state-of-the-art in terms of 1080p video. In other words, in the same bottleneck bandwidth and mobile networks, the average bitrate value of all mobile clients that can be used is 3340kbps in Cubic while all mobile clients in VSiM can at least watch videos with 3613kbps for the average bitrate value regarding 1080p. This is because VSiM considers the mobility pattern and HTTP/3 characteristics (such as server push) to optimize the bandwidth allocation for different mobile users.

• *CDF Comparison.* Fig. 17 and 18 illustrates the CDF of QoE fairness improvement over Cubic, Minerva, GreedyMSMC, and PreCache with ten mobile clients collected by 20 runs in a real wireless network at 15Mbps and 25Mbps bandwidth, respectively. We can see that VSiM outperforms the state-of-the-art with a varied bandwidth. Specifically, as we discussed in Section IV-A, VSiM can accomplish a video quality jump from 720p to 1080p if the improvement with QoE normalization is greater than 5.8 points. We notice that there are $\sim$ 55% (about 6.7 points), $\sim$ 40% (about 6.5 points), $\sim$ 30% (about 6.3 points), and $\sim$ 25% (about 6.4 points) probability for VSiM to achieve the value of QoE fairness improvement being larger than 5.8 points compared to Minerva, PreCache, Cubic, and GreedyMSMC. VSiM fulfills a video quality jump from 720p to 1080p with these probabilities in Fig. 18. For example, suppose the minimum video quality of Minerva over all clients is 720p. In that case, the minimum video quality of VSiM over all clients has a probability of $\sim$ 55% to fulfill 1080p in the same mobile bottleneck environment. This significant improvement depends on the key designed techniques in VSiM for a mobile environment.

• *QoE fairness comparison.* Fig. 19 records the clients with minimum QoE for each run. The longer the box, the greater the variance of the experimental results of different runs, which means that the results are worse. As expected, the lowest bandwidth has the lowest QoE, and vice versa. Besides, we

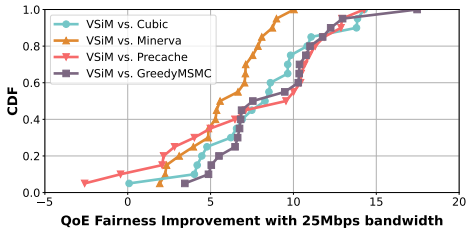


Figure 18: *QoE fairness improvement achieved by clients under various algorithms with 25Mbps bandwidth.*

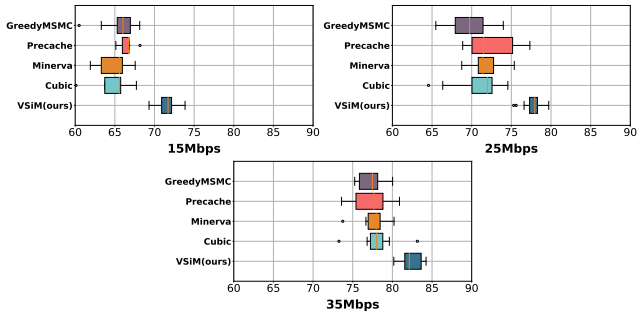


Figure 19: *VSiM fulfills a significant improvement of QoE fairness comparison with various algorithms.*

observe that VSiM achieves good QoE fairness over varying bandwidths. For example, in Fig. 19, the QoE fairness for the median value of VSiM improves an average of about 40% (about 5.9 points with QoE normalization) for all the bandwidth than that of Minerva, which means that VSiM can accomplish a jump from 720p to 1080p. Especially for 15M bandwidth, the median QoE fairness of VSiM improves about 51% (about 6.9 points with QoE normalization) than Cubic. Compared to Minerva, GreedyMSMC, and PreCache, VSiM fulfills about 49% (about 6.6 points), 43% (about 5.9 points), and 36% (about 5.0 points) QoE fairness improvement on the average with aspect to the median value overall bandwidth. Additionally, we observe that the variance (*e.g.*, box size) of VSiM is small over different bandwidths.

Key takeaways. Key takeaways of our evaluations are:

• SDFR server push approach that fulfills the minimum QoE in VSiM is about 2.4 points (equal to clients' viewing experience jump from the bitrate level 2944kbps to 3340kbps in resolution 1080p) (Fig. 9).

• VSiM is robust for heterogeneous wireless networks, including various topologies (Fig. 12), various video lengths and clients scale (Fig. 13), various ABR algorithms and mobility patterns (Fig. 14).

• VSiM improved more than 40% QoE fairness (equal to resolution improvement of clients' viewing quality from 720p to 1080p) compared to state-of-the-art while ensuring about 20% improvement on average for the averaged total QoE (Fig. 16 and 19).

VI. Related Works

We broadly classify the related video streaming optimization literature into the following two categories.

Single user: Huang et al. [44] proposed a method to improve the QoE. Mao et al. [45] presented a system Pensieve, which

trains a neural network model to select future video chunks based on the current environment state. Dong et al. [46] proposed an online-learning congestion control algorithm called PCC Vivace to improve video streaming performance. [7] attempted to optimize QoE by selecting the optimal initial video segment using deep reinforcement learning according to the network conditions (*e.g.*, signal strength). To improve QoE, [8] proposed to integrate the video super-resolution algorithm into the adaptive video streaming strategy by using the deep reinforcement learning approach.

Multiple users: A simple fairness definition would be to provide connection-level fairness, which ensures an equal allocation of network resources among competing flows [34], [47]. In this regard, Jiang et al. [48] proposed an algorithm to improve the fairness, while methods in [49], [50] try to ensure the QoE fairness for competing for flows by exploiting TCP-based bandwidth sharing. [51] proposed a method based on game theory to avoid selfish behavior, which achieves stable viewer QoE during video streaming. Vikram et al. [22] built a system named Minerva, which optimizes max-min QoE fairness by taking into consideration users' priorities in wired networks. QoE optimization is achieved by leveraging the load balance in base stations in [11], where Jain's fairness is used to achieve the bitrate-level fairness for the video streaming traffic. [5] predicted the next base station for mobile clients by using their mobility information and pre-store video in the next base station's cache to achieve video quality consistency. [12] considered the video content, playing buffer, and channel status to optimize the QoE and achieve buffer-level fairness for HTTP adaptive streaming applications. Inspired by the congestion control of transmission control protocol, [13] considered the buffer filling rate, network capacity, congestion avoidance, and detection to optimize the QoE and QoE fairness.

However, single user-based work [7], [8], [44]–[46] did not consider the QoE fairness for the optimization. Besides, they are not suitable for multi-user scenarios with constrained bandwidth. For the multiple-user-based work, [50] and [22] are designed to work optimally for only wired networks. [22], [48]–[50] are not suitable for mobile environments. [5] did not consider QoE fairness of clients. Other works [11], [12], and [13] only considered the specific fairness for clients, such as buffer-level or bitrate-level. Besides, all of them did not incorporate clients' mobility profiles into the QoE fairness optimization for video streaming applications. However, clients have various mobile patterns in wireless networks, which can assist providers in improving clients' QoE and QoE fairness in a high-mobility environment.

VII. Conclusion

In this paper, we propose VSiM, the end-to-end QoE fairness scheme for mobile video traffic with multiple mobile clients. VSiM leverages clients' mobility profiles, QoE-related information, and SDFR server push strategy to allocate bandwidth that maximizes the QoE fairness in real-time. VSiM is easy to deploy in the real world without touching the underlying network infrastructure. We implement VSiM in both simulation and prototype tests on top of HTTP/3. In the simulation, we verify the contribution of each key technique and robustness of VSiM, like different topologies, different video lengths, various mobility patterns, as well as various clients number and ABR algorithms. In the prototype, we find that VSiM outperforms state-of-the-art approaches, with about 40% QoE fairness improvement (equal to clients' viewing experience in resolution from 720p to 1080p). Meanwhile, VSiM ensures about 20% improvements on average of the averaged QoE (equal to the bitrate level improvement of clients' viewing experience from 2087kbps to 2409kbps in 1080p resolution over the public dataset). In future work, we plan to test and deploy VSiM in real-world service provider networks.

References

[1] P. Ramírez-Correa, F. J. Rondán-Cataluña, J. Arenas-Gaitán, and F. Martín-Velicia, "Analysing the acceptation of online games in mobile devices: An application of utaut2," *Journal of Retailing and Consumer Services*, vol. 50, pp. 85–93, 2019.

[2] Z. Su, Q. Xu, F. Hou, Q. Yang, and Q. Qi, "Edge caching for layered video contents in mobile social networks," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2210–2221, 2017.

[3] "Ericsson report: 5g share of mobile data traffic growing," https://www.ericsson.com/49d3a0/assets/local/reports-papers/mobility-report/documents/2022/ericsson-mobility-report-june-2022.pdf, 2022, online; accessed February 11, 2020.

[4] H. Riiser, T. Endestad, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Video streaming using a location-based bandwidth-lookup service for bitrate planning," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 8, no. 3, pp. 1–19, 2012.

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

[6] J. G. Andrews, "Seven ways that hetnets are a cellular paradigm shift," *IEEE communications magazine*, vol. 51, no. 3, pp. 136–144, 2013.

[7] H. Wang, K. Wu, J. Wang, and G. Tang, "Rldish: Edge-assisted qoe optimization of http live streaming with reinforcement learning," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020.

[8] Y. Zhang, Y. Zhang, Y. Wu, Y. Tao, K. Bian, P. Zhou, L. Song, and H. Tuo, "Improving quality of experience by adaptive video streaming with super-resolution," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020.

[9] S. Altamimi and S. Shirmohammadi, "Qoe-fair dash video streaming using server-side reinforcement learning," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 16, no. 2s, pp. 1–21, 2020.

[10] I. Triki, M. Haddad, R. El-Azouzi, A. Feki, and M. Gachaoui, "Context-aware mobility resource allocation for qoe-driven streaming services," in *2016 IEEE Wireless Communications and Networking Conference*. IEEE, 2016, pp. 1–6.

[11] A. Mehrabi, M. Siekkinen, and A. Ylä-Jääski, "Joint optimization of qoe and fairness through network assisted adaptive mobile video streaming," in *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2017, pp. 1–8.

[12] S. Cicalo, N. Changuel, V. Tralli, B. Sayadi, F. Faucheux, and S. Kerboeuf, "Improving qoe and fairness in http adaptive streaming over lte network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 12, pp. 2284–2298, 2015.

[13] M. Seufert, N. Wehner, P. Casas, and F. Wamser, "A fair share for all: Novel adaptation logic for qoe fairness of http adaptive video streaming," in *2018 14th International Conference on Network and Service Management (CNSM)*. IEEE, 2018, pp. 19–27.

[14] L. Li, K. Xu, T. Li, K. Zheng, C. Peng, D. Wang, X. Wang, M. Shen, and R. Mijumbi, "A measurement study on multi-path tcp with multiple cellular carriers on high speed rails," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, 2018, pp. 161–175.

[15] S.-H. Lin, Y. Xu, and J.-Y. Wang, "Coverage analysis and optimization for high-speed railway communication systems with narrow-strip-shaped cells," *IEEE Transactions on Vehicular Technology*, 2020.

[16] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, 2020.

[17] J. Yuen, K.-Y. Lam, and E. Chan, "A fair and adaptive scheduling protocol for video stream transmission in mobile environment," in *Proceedings. IEEE International Conference on Multimedia and Expo*, vol. 1. IEEE, 2002, pp. 409–412.

[18] S. Rosen, B. Han, S. Hao, Z. M. Mao, and F. Qian, "Push or request: An investigation of http/2 server push for improving mobile performance," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 459–468.

[19] H. T. Le, T. Nguyen, N. P. Ngoc, A. T. Pham, and T. C. Thang, "Http/2 push-based low-delay live streaming over mobile networks with stream termination," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2423–2427, 2018.

[20] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A survey on quality of experience of http adaptive streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2014.

[21] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 325–338.

[22] V. Nathan, V. Sivaraman, R. Addanki, M. Khani, P. Goyal, and M. Alizadeh, "End-to-end transport for video qoe fairness," in *Proceedings of the ACM Special Interest Group on Data Communication*, 2019, pp. 408–423.

[23] T. Matsumoto, K. Goto, and M. Yamamoto, "On fairness issue of abr and tcp algorithms in video streaming," *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*, pp. 1–2, 2020.

[24] J. Summers, T. Brecht, D. Eager, and A. Gutarin, "Characterizing the workload of a netflix streaming video server," in *2016 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, 2016, pp. 1–12.

[25] H. Nam, B. H. Kim, D. Calin, and H. Schulzrinne, "A mobile video traffic analysis: Badly designed video clients can waste network bandwidth," in *2013 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2013, pp. 506–511.

[26] K. Nagaraj, D. Bharadia, H. Mao, S. Chinchali, M. Alizadeh, and S. Katti, "Numfabric: Fast and flexible bandwidth allocation in datacenters," in *Proceedings of the 2016 ACM SIGCOMM Conference*, 2016, pp. 188–201.

[27] X. Ge, J. Ye, Y. Yang, and Q. Li, "User mobility evaluation for 5g small cell networks based on individual mobility model," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 528–541, 2016.

[28] X. Lin, R. K. Ganti, P. J. Fleming, and J. G. Andrews, "Towards understanding the fundamentals of mobility in cellular networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 4, pp. 1686–1698, 2013.

[29] B. Gavish and S. Sridhar, "The impact of mobility on cellular network configuration," *Wireless Networks*, vol. 7, no. 2, pp. 173–185, 2001.

[30] F. Bai, N. Sadagopan, and A. Helmy, "Important: A framework to systematically analyze the impact of mobility on performance of routing protocols for adhoc networks," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)*, vol. 2. IEEE, 2003, pp. 825–835.

[31] J. Manner, A. L. Toledo, A. Mihailovic, H. L. V. Munoz, E. Hepworth, and Y. Khouaja, "Evaluation of mobility and quality of service interaction," *Computer Networks*, vol. 38, no. 2, pp. 137–163, 2002.

[32] J. Wang, Y. Zheng, Y. Ni, C. Xu, F. Qian, W. Li, W. Jiang, Y. Cheng, Z. Cheng, Y. Li, X. Xie, Y. Sun, and Z. Wang, "An active-passive measurement study of tcp performance over lte on high-speed rails," *The 25th Annual International Conference on Mobile Computing and Networking*, 2019.

[33] J. Van Der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, T. Bostoen, and F. De Turck, "An http/2 push-based approach for low-latency live streaming with super-short segments," *Journal of Network and Systems Management*, vol. 26, no. 1, pp. 51–78, 2018.

[34] 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.

[35] "Neural networks for multi-output regression," https://machinelearningmastery.com/deep-learning-models-for-multi-output-regression/, 2020, online; accessed January 07, 2022.

[36] "dash.js," https://github.com/Dash-Industry-Forum/dash.js, 2019, online; accessed April 15, 2022.

[37] "A quic implementation in pure go," https://github.com/lucas-clemente/quic-go, 2020, online; accessed May 09, 2021.

[38] S. Lederer, C. Müller, and C. Timmerer, "Dynamic adaptive streaming over http dataset," in *Proceedings of the 3rd multimedia systems conference*, 2012, pp. 89–94.

[39] "Railway mobility model," https://en.wikipedia.org/wiki/Metronom_Eisenbahngesellschaft, 2018, online; accessed August 15, 2021.

[40] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Ya Xu, and Haobo Yu, "Advances in network simulation," *Computer*, vol. 33, no. 5, pp. 59–67, May 2000.

[41] "3gpp tr 38.901 v16.0.0," https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2412, 2019, online; accessed February 11, 2020.

[42] "3gpp tr 36.931 v13.0.0," https://www.arib.or.jp/english/html/overview/doc/STD-T104v4_20/5_Appendix/Rel13/36/36931-d00.pdf, 2016, online; accessed February 11, 2020.

[43] Y. Hao, M. Chen, L. Hu, J. Song, M. Volk, and I. Humar, "Wireless fractal ultra-dense cellular networks," *Sensors*, vol. 17, no. 4, p. 841, 2017.

[44] T. 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 *In Proc. of ACM SIGCOMM*. ACM, 2014, pp. 187–198.

[45] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *In Proc. of ACM SIGCOMM*, 2017, pp. 197–210.

[46] M. Dong, T. Meng, D. Zarchy, E. Arslan, Y. Gilad, B. Godfrey, and M. Schapira, "PCC vivace: Online-learning congestion control," in *In Proc. of USENIX NSDI*, 2018.

[47] M. Allman, V. Paxson, and E. Blanton, "Rfc 5681: Tcp congestion control," International Computer Science Institute, Networking and Security Group, Tech. Rep., 2009.

[48] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with FESTIVE," in *In Proc. of CoNext*, 2012, pp. 97–108.

[49] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and adapt: Rate adaptation for HTTP video streaming at scale," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 4, pp. 719–733, 2014.

[50] X. Yin, M. Bartulovic, V. Sekar, and B. Sinopoli, "On the efficiency and fairness of multiplayer http-based adaptive video streaming," in *In Proc. of IEEE ACC*, 2017.

[51] A. Bentaleb, A. C. Begen, S. Harous, and R. Zimmermann, "Want to play dash?: a game theoretic approach for adaptive streaming over HTTP," in *In Proc. of ACM MMSys*, P. César, M. Zink, and N. Murray, Eds., 2018.

VIII. Appendix

A. Convergence Proof

In the following part, we prove that the bandwidth allocation method in Sec. III-B will converge to utility fairness.

There are n clients in a mobile network and all clients share the same bottleneck bandwidth to download videos from a server. The egress bandwidth of the server is fixed and it is denoted by B. The i^{th} client has a utility function $U_i\left(r_i\right)$ in which r_i denotes its available bandwidth. We wish to find a fair bandwidth allocation for the client with the intention to maximize the QoE of the clients with minimum QoE.

It is reasonable to say that the available bandwidth for client c_i can be B at most, thus, we could get

$$\arg\max_{r_i} U_i(r_i) = B. \tag{4}$$

Therefore, we could normalize the utility function as follows:

$$\widetilde{U_i}\left(r_i\right) = \begin{cases} 0, & r_i = 0, \\ \dfrac{U_i\left(r_i\right)}{U_i\left(B\right)}, & 0 < r_i < B, \\ 1, & r_i = B. \end{cases} \tag{5}$$

With the above utility function definition, our optimization problem can be modeled as

$$\max \quad \min_i \widetilde{U_i}(r_i) \tag{6a}$$

$$\text{s.t.} \quad \sum_i r_i = B. \tag{6b}$$

It is reasonable to infer that the utility function is concave since clients' experience diminishes the marginal utility as the bandwidth increases. Then, we have the following theorem

Theorem VIII.1. *$\widetilde{U}(r_i)$ is non-decrease concave function, for $0 \le x \le y \le B$, $0 < \alpha \le 1$ we have*

$$\left(\frac{y}{x}\right)^{\alpha} \le \frac{U(y)}{U(x)} \le \frac{y}{x}. \tag{7}$$

Theorem VIII.2. *There exists an optimal allocation $\{r_i^*\}$ that reaches the goal in which $\{\widetilde{U_i}(r_i)\}$ are equal for all participating clients. At each time window, we could get a series of weights using*

$$w_i = \frac{r_i}{\widetilde{U_i}(r_i)},$$

then we could allocate the egress bandwidth as

$$r_i = \frac{w_i}{\sum_{i=1}^{N} w_i} B.$$

The above allocation ensures that r_i will converge to r_i^* after t iterations.

B. Proof of Theorem VIII.1

Proof. Let $f(t) = \frac{U(t)}{t}$, then we can get

$$f'(t) = \frac{U'(t)\,t - U(t)}{t^2},$$

$$\frac{\partial\left(U'(t)\,t - U(t)\right)}{\partial t} = U''(t) + U'(t) - U'(t) = U''(t) \le 0.$$

Hence, for the term $U'(t)\,t - U(t)$, we know it takes the maximal value at $t = 1$, i.e.,

$$\max_{t\in[0,1]} \left(U'(t)\,t - U(t)\right) = U'(0) * 0 - U(0) \le 0.$$

So for $t \in [0,1]$, $U'(t)\,t - U(t) \le 0$, then $f'(t) \le 0$. Thus, $f(t)$ is decrease function, we then can get $\frac{U(y)}{y} \le \frac{U(x)}{x}$. Thus, we prove

$$\frac{U(y)}{U(x)} \le \frac{y}{x}.$$

Similarly, we can prove the left side and thus, the theorem is proved. □

C. Proof of Theorem VIII.2

Proof. We first prove the convergence for special case, i.e., for two clients. We denote the two clients' utility functions as $\widetilde{U_1}$ and $\widetilde{U_2}$, respectively. The bandwidth of client i in t iteration is denoted by r_i^t. There exists an optimal bandwidth allocation (r_1^*, r_2^*) satisfying the condition that $\widetilde{U_1}(r_1^*) = \widetilde{U_1}(r_2^*)$.

Without loss of generality, we hope to prove the convergence that $r_1^t \to r_1^*$ and $r_2^t \to r_2^*$. It is equivalent to prove that $\frac{r_2^t}{r_1^t} \to \frac{r_2^*}{r_1^*}$.

In each iteration of the weight updates, if the clients compute their weights $w_i = \frac{r_i}{\widetilde{U_i}(r_i)}$, then we could get

$$\frac{r_2^{t+1}}{r_1^{t+1}} = \frac{w_2}{w_1} = \frac{\widetilde{U_1}(r_1^t)}{\widetilde{U_2}(r_2^t)} \frac{r_2^t}{r_1^t}. \tag{8}$$

.

We denote $X_1^t = \frac{r_1^*}{r_1^t}$ and $X_2^t = \frac{r_2^t}{r_2^*}$, from Equation (8), we could get

$$X_1^{t+1} X_2^{t+1} = \left(\frac{\widetilde{U_1}(r_1^t)}{\widetilde{U_1}(r_1^*)} X_1^t\right)\left(\frac{\widetilde{U_2}(r_2^*)}{\widetilde{U_2}(r_2^t)} X_2^t\right). \tag{9}$$

On the other hand, from Theorem (VIII.1), we could get

$$\frac{r_1^t}{r_1^*} \le \frac{\widetilde{U_1}(r_1^t)}{\widetilde{U_1}(r_1^*)} \le \left(\frac{r_1^t}{r_1^*}\right)^{\alpha}, \tag{10}$$

$$\left(\frac{r_2^*}{r_2^t}\right)^{\alpha} \le \frac{\widetilde{U_2}(r_2^*)}{\widetilde{U_2}(r_2^t)} \le \left(\frac{r_2^*}{r_2^t}\right). \tag{11}$$

Then we could get

$$1 \le X_1^{t+1} X_2^{t+1} \le \left(X_1^t X_2^t\right)^{1-\alpha} \le \left(X_1^0 X_2^0\right)^{(1-\alpha)t}. \tag{12}$$

As $t \to 1$, $X_1^{t+1} X_2^{t+1} = 1$, then we can conclude that $r_1^t \to r_1^*$ and $r_2^t \to r_2^*$.

The above procedures ensure that the proposed bandwidth allocation method could realize fairness in the end. Using the above procedures recursively we can conclude that VSiM will allocate bandwidth fairly i.e., optimally in terms of utility. Thus, Theorem (VIII.2) is proved. □

D. Video dataset in our paper

Level	Bitrate(kbps)	Resolution	Level	Bitrate(kbps)	Resolution
V_1	45	320*240	V_{11}	782	1280*720
V_2	88	320*240	V_{12}	1008	1280*720
V_3	128	320*240	V_{13}	1207	1280*720
V_4	177	480*360	V_{14}	1473	1280*720
V_5	217	480*360	V_{15}	2087	1920*1080
V_6	255	480*360	V_{16}	2409	1920*1080
V_7	323	480*360	V_{17}	2944	1920*1080
V_8	378	480*360	V_{18}	3340	1920*1080
V_9	509	854*480	V_{19}	3613	1920*1080
V_{10}	577	854*480	V_{20}	3936	1920*1080

Yali Yuan received her Ph.D. degree from Göttingen University, Göttingen, Germany, in 2018. Dr. Yuan joined the School of Cyber Science and Engineering, Southeast University, Nanjing, China, as an assistant professor in 2021. Her research interests include intelligent network and network traffic analysis.

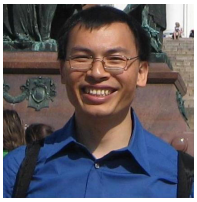
Weijun Wang respectively received the Ph.D. degrees from University of Göttngen, Germany and Nanjing University, China. Now, he is a Postdoc Researcher at Institute for AI Industry Research, Tsinghua University. Before that, he served as a research employee at University of Göttngen from 2020-2023. He is a member of IEEE and ACM.

Yuhan Wang received his Bachelor's degree from East China University of Science and Technology, China in 2016, and Master's degree from the University of Göttngen, German in 2021. His research interests include intelligent network and 5G/6G networks.

Sripriya Srikant Adhatarao received her M.Sc. and Ph.D. in computer science from the Computer Networks Group, Institute of Computer Science, University of Göttingen. She is currently working as a Senior researcher in Huawei Munich Research Center. Her research areas of interest include topics such as mobile networks (especially 5G and 6G), machine learning, multi-connectivity and edge.

Bangbang Ren received his Ph.D. degree, Bachelor's and Master's degrees from the National University of Defense Technology, China, in 2021, 2015, and 2017, respectively. He was also a visiting research scholar of the University of Göttingen, German, in 2019. His research interests include software-defined network and network optimization.

Kai Zheng (Senior Member, IEEE) is currently the Director of the Computer Network and Protocol Research Laboratory, Huawei Technologies. His research interests include architectures and protocols for the next generation networks, such as 5G/IoT networks, cloud oriented data center networks, RDMA networks, and real-time multimedia networks.



Xiaoming Fu (M'02-SM'09-F'22) received the Ph.D. degree in computer science from Tsinghua University, China, in 2000. He is a Professor and the Head of the Computer Networks Group, University of Göttingen. He has also held visiting positions at ETSI, University of Cambridge, Columbia University, Tsinghua University, and UCLA. He is a fellow of IEEE , a distinguished member of ACM, a fellow of IET, and a member of Academia Europaea.