Towards Smart Routing: Exploiting User Context for Video Delivery in Mobile Networks

Jun He and Wei Song
Faculty of Computer Science
University of New Brunswick, Fredericton, Canada
Emails: {jhe2, wsong}@unb.ca

***Abstract*—Mobile networks are undergoing active enhancement and fast evolution, so as to host the ever-growing data traffic, mainly fueled by video services. Despite ongoing efforts to improve the last-hop transmission in Radio Access Networks (RANs), traffic scheduling and routing in core networks remain challenging. In a system swamped with video requests, the core network needs first to schedule the transmission rate for each request, then to redirect requests to respective source nodes, and finally to route so-determined peer-to-peer flows. Towards smart routing, this paper focuses on the following two problems: (1) how to manage Quality of Experience (QoE) of video streaming services; and (2) how to optimize request routing in the core network. We exploit user context and formulate a joint problem simultaneously addressing these problems. We analyze the hardness of the formulated problem and propose a fast approximate routing algorithm, which adaptively schedules transmission rate and strategically routes the scheduled video demands. Theoretical analysis and computer simulations are then carried out to study the efficiency of the proposed algorithm.**

***Index Terms*—Video request routing, user context, software-defined mobile core network, source virtualization, smart routing.**

I. Introduction

Global mobile data traffic is expected to explode 11-fold by 2018, predominantly fueled by video traffic [1]. To cope with the growth, components and protocols of current mobile networks, *e.g.,* the Long Term Evolution (LTE) systems, are enhanced or re-factored by integrating transparent video caching and implementing more efficient request-routing algorithms [2]–[5], while mobile systems are undergoing fast evolution from current fourth generation (4G), *i.e.,* LTE/LTE-Advanced, to the fifth generation (5G) [6] with virtualized networking functions [7]. Various video sources, such as in-system caching [3], Mobile Content Distribution Networks (MCDNs) [8], and cloud service points [9] will be deployed in core networks so as to move data close to customers.

To promote manageability and flexibility, the control plane is separated from the data plane in LTE systems and evolving mobile systems, *i.e.,* control information is delivered in the control plane while data traffic is transferred in the data plane, enabling centralized networking management in core networks. Fig. 1 depicts the infrastructure of the mobile networks studied in this paper. The mobile core consists of a central controller, node sets of internal content sources, intermediate forwarding servers and edge servers. Edge servers refer to these servers at the edge of the mobile core, providing network service to Radio Access Networks (RANs). The solid lines therein represent the data plane, while the dashed lines represent the control plane. In the mobile core network, the central controller periodically gathers network and traffic information, and accordingly shapes network flows and optimizes the network performance. The infrastructure is an instant abstraction of future mobile networks with SDN cores [10]. By mapping the controller to the PDN gateway, edge servers to serving gateways, intermediate servers to routers, we note that it can also be considered as an enhanced LTE system with built-in content storage.

This research was supported by Mariner Partners Inc. and Atlantic Canada Opportunities Agency, Canada.

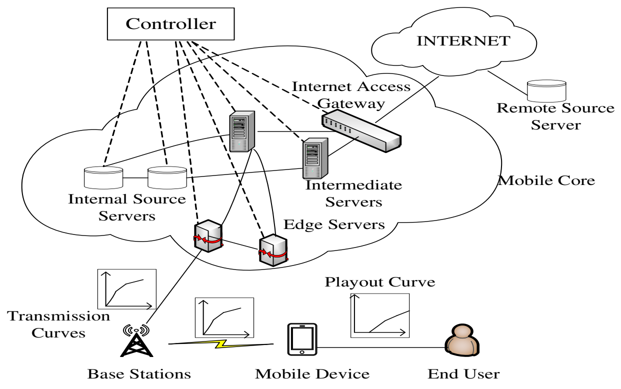

Figure 1. Video delivery over enhanced/evolving mobile networks.

Video requests are initiated by mobile devices, while entering the core network at edge servers. Request routing in the mobile core network is therefore to determine how to deliver requested video streams to corresponding edge servers. Each video request can potentially be fulfilled by multiple sources through multiple paths inside the network. After gathering request information from edge servers, the controller needs to route these requests. Specifically, request routing hereby makes decisions on both *source redirection*, *i.e.,* how to redirect requests to respective sources, and *flow routing*, *i.e.,* how to route so-generated peer-to-peer flows in the network.

Modern mobile devices equipped with advanced hardware generally have plenty of memory for buffering video streams. For a particular mobile device, the user context here stands for its buffer information as well as the way the corresponding user consumes data. As shown in Fig. 1, transmission curves represent how edge servers deliver data, while playout curves indicate how the data is actually consumed. Detailed information of these curves is shown in Fig. 2, where x-axis is the time and y-axis is the aggregate amount of data received or consumed. The

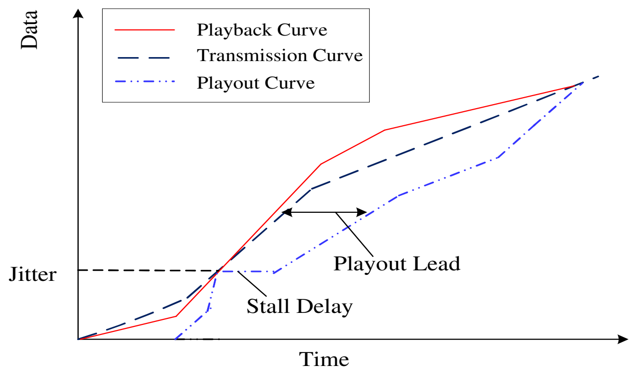

Figure 2. Playback, playout and transmission curves of a video stream.

playback curve is a characteristic of corresponding video and is independent of underlying transmission and user context. With buffer, temporary failure of transmission does not necessarily cause jitters or stalls. Stalls happen only when the buffering data is not able to support normal playback. Depending on the data transmission and the stall-recovery scheme, a stall incurs certain duration of *stall delay*.

The term of *playout lead* is adopted from [11], and represents the duration of time the video can be played using only the data already buffered in the mobile device. The playout lead plays a critical role in determining whether video playback stalls, thus is a key factor to manage the quality of experience (QoE) of mobile customers. In [11], Dutta *et al.* attempted to prevent stalls by maximizing the minimum playout lead. Likewise, Liang *et al.* [12] tried to reduce stall delay by adapting variable bit-rate of video streams to dynamic wireless channels and user buffer. These studies are conducted towards a single-hop wireless network while assuming all requested data has been pre-fetched and stored at base stations. On the contrary, this work completes the application of existing works to mobile networks by removing the pre-fetching assumption and attempts to adapt transmission rate of requests to QoE demands depending on respective user context.

In summary, our contributions are three-fold. Firstly, to the best of our knowledge, we are the first to exploit user context for video delivery in mobile core networks. As one of the key features of future mobile networks [6], a route of adapting network behavior to user context is sketched out in this work. The problem and the system model we consider here are essential to a robust and high-performance core system in various mobile networks evolving to 5G. Secondly, we formulate an optimization framework which jointly considers transmission data adaptation, source redirection and flow routing. We aim to manage QoE of customers by maximizing minimum playout lead over the whole network while maintaining the maximum link utilization under a given threshold. Thirdly, we analyze the hardness of the formulated problem and propose a fast algorithm for it with a provably approximate ratio and tolerable computing overhead as well as negligible traffic overhead.

The rest of this paper is organized as follows. In Section II, we describe the system model, propose the optimization framework and analyze the hardness of the formulated problems. Section III gives our solutions to the problems formulated. Simulation studies are presented in Section IV. Finally, Section V concludes the paper.

II. System Model and Formulations

A. System Model

We consider an evolving mobile core network as depicted in Fig. 1. Video requests are initiated by mobile devices and reach the core network through corresponding edge servers. Every video request can be fulfilled by one or more sources in the core network. We assume that the controller has complete static information of the core network, and is able to periodically collect dynamic load information and video requests from the control plane. The controller repeats request optimization from interval to interval. Different from [11] where the interval should be small enough to catch up with the dynamics of wireless channel condition, here in core networks, we consider a much greater duration of interval in a time scale comparable to the dynamics of user's engagement, typically a few seconds [13]. Within each period of interval, the request patterns are assumed to be fixed, and related information, *e.g.,* video clips, remote buffer and playing time, is assumed to be available.

Conventional routing schemes target optimizing certain network metric by selecting paths for fixed peer-to-peer flow demands. In our model, the controller also needs to solve the source-redirection problem, *i.e.,* how to select sourcing nodes to fulfill each request. Towards smart routing, we further consider the adaptation of video demands depending on respective playing time and remote buffer.

The mobile network is formulated as a directed graph $G = (N, V, E)$, where N is the set of video clips (or interchangeable chunks) with $n = |N|$, V is the set of interconnected networking nodes with $v = |V|$, and E is the set of links with $m = |E|$. These networking nodes include intermediate servers, edge servers as well as data-source servers. Link $e \in E$ has a capacity $c(e)$, of which up to λ_0 fraction can be used for video delivery. For every pair of nodes $i, j \in V$, let $\mathcal{P}_{ij}$ be the set of paths from i to j, and $\mathcal{P} = \cup_{(i,j)}\mathcal{P}_{ij}$ be the union of all path sets. Moreover, let $\mathcal{P}_e$ be the set of all paths in $\mathcal{P}$ that use edge e for all $e \in E$.

Regarding video requests, we define S_k as the set of sourcing nodes capable of serving requests for video clip $k \in N$. The set of requests is denoted by R . At the remote side, for a request $(i, k) \in R$ at edge node i for video k, the context information, *i.e.,* current playing time t_i^k and current buffer size of b_i^k, is assumed to be gathered or estimated by the controller. The transmission data rate of request (i, k) scheduled for the following duration of $\triangle t$ consists of two parts, *i.e.,* a fixed lowest data rate l_i^k and an adjustable data rate of d_i^k to be discussed later. We further use $\mathcal{P}_i^k$ to denote the set of all available paths to serve request $(i, k) \in R$, *i.e.,* $\mathcal{P}_i^k = \cup_{j\in S_k}\mathcal{P}_{ji}$.

We now draw attention to request routing optimization for each time interval of $\triangle t$, in which we need to select the adjustable data rate d_i^k for request (i, k), assign multiple requests to sources and route peer-to-peer video streams.

B. Objectives and Optimization Framework

The playout lead of a request represents the duration of extra time the remote user can play the video using only its buffer data. Accordingly, the playout lead of request (i,k) at the end of current scheduling interval of $\triangle t$, *i.e.,* $L(i,k)$, is defined as follows:

$$L(i,k) = q_k(b_i^k + d_i^k \triangle t + l_i^k \triangle t) - t_i^k - \triangle t \tag{1}$$

where $q_k(\cdot)$ is the playout function of video k, mapping aggregate amount of data to playout time (see Fig. 2). Different from [11], the lead definition here also includes the minimum data rate l_i^k. In other words, we need to maintain certain lowest data rate for all requests even though some of them can have greater playout lead. This aims to prevent scheduling starvation for old sessions with great lead values.

At the beginning of an interval, we route requests aiming to maximize the minimum playout lead by the end of the interval, while keeping the maximum link usage under a given threshold. The problem is then formulated as a linear programming (LP) problem:

$$\max \quad \min_{(i,k)\in R} L(i,k) \tag{2}$$

$$\text{s. t.} \quad \sum_{P:e\in P} x(P) \le \lambda_0 c(e),\ \forall e \in E \tag{3}$$

$$\sum_{P\in\mathcal{P}_i^k} x(P) \ge d_i^k + l_i^k,\ \forall (i,k)\in R \tag{4}$$

$$x(P) \ge 0, \qquad \forall P \in \mathcal{P} \tag{5}$$

where $L(i,k)$ in the objective is defined in (1), the first constraint indicates the link capacity limitation, scaled by factor λ_0, and the second constraint means the demand for request (i,k) needs to be fulfilled. We also note that the variables herein include path flow values $\{x(P)\}$ as well as the adjustable rates $\{d_i^k\}$.

Before carrying further discussion on the problem (2), we first review the traffic engineering (TE) problem studied in our previous work [14], in which we attempt to minimize the maximum link utilization, *i.e.,* the ratio of traffic load to link capacity. The TE problem is expressed as:

$$\min \quad \lambda \tag{6}$$

$$\text{s. t.} \quad \sum_{P:e\in P} x(P) \le \lambda c(e),\ \forall e \in E \tag{7}$$

with (4) and (5). Here, the first constraint indicates the link capacity limitation, scaled by the utilization factor λ. We note that the optimal value of the TE problem is proportional to the attached traffic load, *i.e.,* $d_i^k + l_i^k$.

The TE problem is a reduced case of problem (2), where transmission data rates are fixed. If there exists an oracle for selecting data rate set $\{d_i^k\}$ for the next routing interval, solving the TE problem is ready to determine whether the selected rate set is routable under the maximum link-usage limitation, *i.e.,* λ_0. However, the TE problem has a prohibitively large size for any direct solution via modern LP solvers [14]. Likewise, problem (2) is of similar complexity. Therefore, a fast algorithm with approximation guarantees is more desirable.

III. The Approximate Approach

A. The Approximate Algorithm for the TE Problem

The traffic engineering problem formulated in (6) has been extensively studied in our previous work [14]. For completeness, we here list the critical parts of the algorithm from [14].

We first formulated an equivalent formulation of problem (6) as follows:

$$\begin{aligned} \max \quad & \pi \\ \text{s. t.} \quad & \sum_{P:e\in P} y(P) \le c(e), \quad \forall e\in E \\ & \sum_{P\in\mathcal{P}_i^k} y(P) \ge \pi d_i^k, \quad \forall (i,k)\in R \\ & y(P) \ge 0, \quad \forall P \in \mathcal{P}. \end{aligned} \tag{8}$$

In the formulation above, we scaled the transmission data rate of d_i^k by factor π, and attempted to maximize such a factor. The equivalence of the TE problem and problem (6) is clear: by letting $x(P) = y(P)/\pi$ and $\lambda = 1/\pi$.

We then showed the similarity of problem (8) with the maximum concurrent flow problem in the category of multi-commodity flow (MCF) problems [15]. While the conventional MCF problem identifies a commodity by its source-destination pair, such source-destination pairs need to be determined in problem (8). Problem (8) can be viewed as a variant of the maximum concurrent flow problem with multiple sources and one destination for each commodity. Such a variant was first studied in [14], where we leveraged the results in [15] and proposed $(1+\omega)$-approximation algorithm for it for any $\omega > 0$. Detailed proofs and discussion can be found in [14]. The main results therein are listed as follows:

Theorem 1. *There exists a fully polynomial-time approximation schemes (FPTAS) for problem (8), which achieves an approximation factor of* $(1+\omega)$*, for any* $\omega > 0$.

Theorem 2. *Define* $r_{\max}$ *to be the maximum number of requests initiated from one node, i.e.,* $r_{\max} = \max_{i\in V} |R_i|$*, and* $T_s = O(v\log v + m + r_{max})$*. The algorithm proposed in [14] computes a* $(1+\omega)$*-approximation solution to problem (8) in* $\tilde{O}((\omega^{-2}+\log r)m\cdot T_s)$ *time, where* $\tilde{O}(f) = O(f\cdot\log^{O(1)} m)$.

Accordingly, the TE problem can be $(1+\omega)$-approximately solved with computational complexity proportional to ω^{-2}, for any $\omega > 0$. Recalling the relation of the TE problem and problem (2) revealed in Section II-B, we can now solve problem (2) in the following via a binary-search scheme.

B. The Binary-Search Scheme

By introducing a bound value α, we can rewrite the max-min objective of problem (2) in the equivalent form below:

$$\max \quad \alpha \tag{9}$$

$$\text{s. t.} \quad L(i,k) \ge \alpha,\ \forall (i,k) \in R \tag{10}$$

with (1), (3), (4), and (5).

Corresponding to each lower bound α selected for the playout lead in (10), there is a set of adjustable data rates $\{d_i^k\}$ computed through the definition of the playout lead in Eq. (1).

On the other hand, for a given data-rate set, the min-max link utility can be estimated by solving a corresponding TE problem. The data-rate set is feasible if the estimated link utility does not exceed the pre-set upper bound λ_0. The feasibility of a lower bound α is therefore verified. The objective of problem (9) is then to find the feasible upper bound for α.

Apparently, we can obtain such an upper bound using binary search. The algorithm follows a stardard binary-search route, *i.e.,* , starting with a feasible initial lower bound α_- and an upper bound α_+, iteratively searching for the greatest feasible α until an accuracy threshold ϵ is reached (ϵ and ω are in the same order). We term this algorithm as the *Binary Algorithm*. Due to space limit, we omit the detailed procedure here and refer the reader to the longer version of this paper [16].

Our algorithm outputs a path-flow solution, in which adjustable data rates are implied. Together with the flow-conversion algorithm in [14], we obtain a set of hop-by-hop routing decisions, which jointly account for the three subproblems listed in Section I: data-rate selection, source server selection and flow routing. We also note that α_+, α_- and α here are not necessarily positive values. For instance, when the system is under extremely heavy load, the proposed algorithm computes a path-flow assignment that minimizes the stall delay of the remote user with the worst buffer condition.

The algorithm includes $\log B$ times of solving the subproblem of traffic engineering, where B is the largest number used to specify binary search with respect to the initial bound setting (α_-, α_+), and the accuracy factor ϵ. Together with Theorem 2, we can find a solution to problem (2) in $\tilde{O}((\omega^{-2} + \log r)m \cdot T_s \cdot \log B)$ time.

According to the procedure of our algorithm, we further have the following results.

Theorem 3. *If problem (9) is optimally solved, then in the solution, the corresponding subproblem of TE also reaches its optimal value λ_{OPT}. Furthermore, we have $\lambda_{OPT} = \lambda_0$.*

Theorem 4. *Let (i', k') be the request with minimum value of playout lead and $\delta = \omega d_{i'}^{k'} \triangle t / r_{i'}^{k'} + \epsilon$. Binart Algorithm computes a δ-suboptimal solution to problem (2).*

Proofs are omitted due to space limit and can be found in [16]. In real systems, the minimum data rate $d_{i'}^{k'}$ produced by Binary Algorithm is not far greater than its playback rate $r_{i'}^{k'}$, *e.g.,* $0 \leq d_{i'}^{k'} / r_{i'}^{k'} \leq 3$. Hence, δ is in the same order as ω and ϵ.

IV. Performance Evaluation

A. Setup

We employ the simulator developed in [14] and implement the rate adaptation algorithms with around 1000 lines of C++ code. We evaluate our algorithms in a core network with 50 nodes including 30 edge nodes and 20 intermediate nodes. There are $200,000$ video clips randomly distributed among the intermediate nodes so that each clip has an average number of 5 replicas. We adopt the single-layer video dataset from [17], where videos are formated in MPEG-4 Variable Bit Rate and play out at a fixed frame rate of 30 frames per second. Video clips are cut into YouTube-like short videos with mean play length of 252 seconds [18]. Data rate can be adjusted from 128 kbps to 1.5 Mbps. The lowest data rate is 128 kbps. The threshold for maximum link utilization λ_0 is 0.9. The default values of ϵ and ω are 0.1 and 0.05, respectively.

We use a hybrid star-ring network, where 49 nodes are interconnected in a ring and each node on the ring has a link to the node at the star center. All links are assumed to be bi-directional with capacity of 1 Gbps for each direction. In the simulations, we control a load factor named *traffic density*, which denotes the average number of requests collected from edge servers at the beginning of each interval.

B. Dynamic Request Routing vs Static-Configuration Routing

Current video systems, *e.g.,* Netflix [19], use static configurations for source redirection. A client-to-source configuration is changed only when the client experiences an over-threshold delay. Flow-level optimization in current systems depends on conventional lower-layer schemes, *e.g.,* the Open Shortest Path First (OSPF) protocol. For comparison, we implement the static-configuration routing scheme in our experiments, where each request is fulfilled by the nearest source, and the so-generated IP traffic is optimized by the OSPF algorithm.

We align the dynamic request routing in this paper with the static-configuration routing, and show the results of maximum link utilization in Fig. 3. From the figure, we can see that the dynamic request routing significantly outperforms the scheme with static configurations. The rationale behind this lies not only in dynamically optimizing flow routing but also in concurrently fulfilling requests from multiple sources.

C. Context-aware Rate Adaption vs. Speculative Schemes

For comparison fairness, we assume that the dynamic request routing is adopted by all remaining experiments in this section, when evaluating the efficiency of context-aware rate adaptation by comparing to speculative schemes.

One baseline scheme for comparison is to transfer data at a fixed rate associated with the corresponding encoding rate, termed as the *Fixed-rate* scheme. This scenario exists in non-buffer streaming service, *e.g.,* live streaming or the dynamic adaptive streaming over HTTP, *i.e.,* DASH [20]. Here we use 1.20 times of the encoding rate. Another speculative scheme is the *Burst & Fixed* strategy, (possibly) adopted by YouTube [18]. Servers commence a download by an initial burst of 40 seconds of data at the maximum available bandwidth the network can support, and apply a throttling algorithm imposing a data rate of 1.25 times of the video encoding rate. These rates are proportionally scaled down if needed to assure the maximum link utilization threshold λ_0.

We run the simulations for 200 intervals. Data rates are selected at the beginning of each interval by corresponding schemes. The length of each interval is 6 seconds. Video requests arrive at edge servers with a Poisson arrival rate of 80 per interval. Here we focus on the occurrence of stalls while assuming that stalls are recovered only by the scheme of fixed buffered playout data [11], *i.e.,* videos resume display after a fixed amount of data is received.

Fig. 4 and Fig. 5 show the number of stalls that happen during the experiments and the average number of active sessions per edge server, respectively. From the figures, we

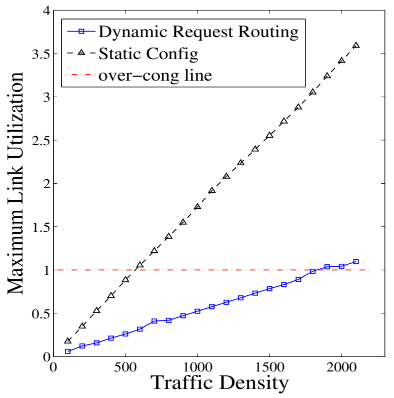

Figure 3. TE results in star-ring network.

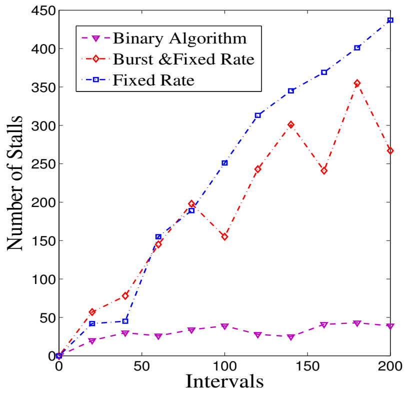

Figure 4. The total number of stalls.

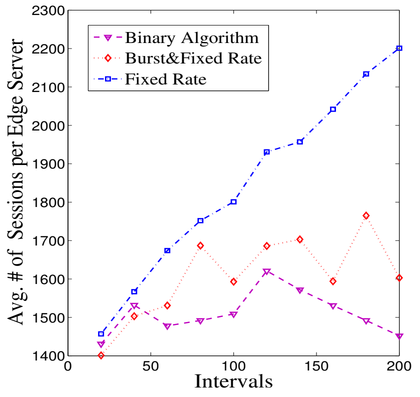

Figure 5. Avg. number of active sessions.

can see that our schemes produce much fewer stalls than the alternative ones. The Burst & Fixed curve experiences large fluctuations due to the burst behaviors (see Fig. 4), while it attempts to finish sessions as quickly as possible, resulting in a lower average number of active sessions (see Fig. 5). On the contrary, the Fixed-rate scheme has no intention to quickly detach sessions. Sessions gradually accumulate (see Fig. 5), producing a high probability of stalls (see Fig. 4).

The schemes of Binary Algorithm start with a higher rate while the rate decreases as the playout lead increases. The final part of the file is delivered with the minimum rate, *i.e.,* $l_i^k = 128$ kbps, in both schemes. The Burst & Fixed scheme aggressively obtains a greater playout lead for the first 40 seconds, while the rate decreases quickly thereafter due to the aggressive behaviors from new sessions in the network. No stalls occur in the above three schemes. The Fixed-rate scheme produces lots of stalls and finishes the delivery at around 300 seconds after the commence of the session. Thus, it produces an aggregate stall delay of $300 - 240 = 60$ seconds.

From these experiments, we conclude that Binary Algorithm can intelligently adapt data rates with respect to network conditions as well as user context, thus efficiently reduce the occurrence of stalls. On the other hand, the non-cooperative behaviors of the Burst & Fixed scheme potentially produce large fluctuations in the network, while the conservative Fixed-rate scheme neglects both network conditions and user context, which results in the worst performance in our simulations.

V. Conclusion

In this work, we have exploited the information of user context to optimize the video delivery in mobile core networks evolving from 4G/LTE to 5G. We formulate a joint problem of data-rate selection, source redirection and flow routing. We manage the QoE of customers by maximizing the minimum playout lead using the static information of video playback curve as well as the dynamic information of estimated remote video buffer and remote playing time. A fast algorithm has been proposed for the formulated problem. Theoretical analysis and computer simulations show the efficiency of the algorithm. We conclude that this study moves an important step towards smart routing, where videos are delivered upon network conditions as well as instant playing demands.

References

[1] Cisco. Cisco visual networking index: Global mobile data traffic forecast update, 2013-2018, February 2014.

[2] F. Z. Yousaf et al. Mobile CDN enhancements for QoE-improved content delivery in mobile operator networks. *IEEE Network*, 27(2):14–21, 2013.

[3] J. He et al. A collaborative framework for in-network video caching in mobile networks. In *Proc. IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON) 2013*, pages 643–724.

[4] J. He, X. Zhao, and B. Zhao. A fast, simple and near-optimal content placement scheme for a large-scale VoD system. In *Proc. IEEE International Conference on Com. Systems (ICCS)*, pages 378–382, 2012.

[5] J. Zhu et al. EPCache: In-network video caching for LTE core networks. In *Proc. Wireless Communications & Signal Processing (WCSP)*, 2013.

[6] ADVA et al. Horizon 2020 advanced 5G network infrastructure for future Internet PPP, draft version 2.1. 2013.

[7] N. Chowdhury and R. Boutaba. A survey of network virtualization. *Computer Networks*, 54(5):862–876, 2010.

[8] M. Liebsch and F. Z. Yousaf. Runtime relocation of CDN serving points-enabler for low costs mobile content delivery. In *Proc. IEEE Wireless Comm. and Networking Conference (WCNC)*, pages 1464–1469, 2013.

[9] J. He and W. Song. Evolving to 5G: A fast and near-optimal request routing protocol for mobile core newtorks. In *Proc. IEEE Globecom*, 2014.

[10] K. Pentikousis, Y. Wang, and W. Hu. Mobileflow: Toward software-defined mobile networks. *IEEE Communications Magazine*, 51(7), 2013.

[11] P. Dutta, A. Seetharam, V. Arya, M. Chetlur, S. Kalyanaraman, and J. Kurose. On managing quality of experience of multiple video streams in wireless networks. In *Proc. IEEE INFOCOM 2012*, pages 1242–1250.

[12] G. Liang and B. Liang. Balancing interruption frequency and buffering penalties in VBR video streaming. In *Proc. IEEE INFOCOM*, pages 1406–1414, 2007.

[13] F. Dobrian et al. Understanding the impact of video quality on user engagement. *ACM SIGCOMM-Computer Comm. Review*, 41(4):362–373, 2011.

[14] J. He and W. Song. Optimizing video request routing in mobile networks with built-in content caching. Technical report, University of New Brunswick, Canada, http://cs.unb.ca/~wsong/optrouting.pdf, March 2014.

[15] G. Karakostas. Faster approximation schemes for fractional multicommodity flow problems. In *Proc. 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 166–173, 2002.

[16] J. He and W. Song. Towards smart routing: Exploiting user context for video delivery in mobile networks. Technical report, University of New Brunswick, Canada, http://cs.unb.ca/~wsong/smart.pdf, May 2014.

[17] P. Seeling et al. Network performance evaluation using frame size and quality traces of single-layer and two-layer video: A tutorial. *IEEE Comm. Surveys & Tutorials*, 6(3):58–78, 2004.

[18] P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz, and J. M Lopez-Soler. Analysis and modelling of YouTube traffic. *Transactions on Emerging Telecommunications Technologies*, 23(4):360–377, 2012.

[19] V. K. Adhikari et al. Unreeling NetFlix: Understanding and improving multi-CDN movie delivery. In *Proc. IEEE INFOCOM, 2012*, pages 1620–1628.

[20] B. Vandalore et al. A survey of application layer techniques for adaptive streaming of multimedia. *Real-Time Imaging*, 7(3):221–235, 2001.