QoE Optimization Based on Quality-delay Trade-off Model for Adaptive Streaming with Multiple VBR Videos

Nguyen Thi Kim Thoa*, Pham Hong Thinh, Pham Ngoc Nam

Hanoi University of Science and Technology – No. 1, Dai Co Viet Str., Hai Ba Trung, Ha Noi, Viet Nam

Received: November 22, 2018; Accepted: November 26, 2018

Abstract

HTTP adaptive streaming (HAS) as a part of multi-bitrate streaming, has attracted significant attention over the past few years. Although offering many advantages such as easy deployment and effective cost, HAS faces some challenges in providing users with high video quality. In managed networks (i.e., IPTV), the purely client-driven approaches of current HAS cause competing behavior, excessive quality oscillations, which negatively affect user experience. Some recent studies have proposed network-based solutions to overcome these problems; however, they just target at constant bitrate (CBR) videos. In this paper, we propose a quality optimization solution for variable bitrate (VBR) video streaming which allows components inside the network to select an appropriate version for each HAS client. The experiments in real-time conditions show that our method can provide each HAS client with the best possible quality while meeting the constraints of overall bandwidth and delay.

Keywords: Adaptive Streaming, QoE, Optimization, Variable bitrate (VBR).

1. Introduction

Recently, multi-bitrate adaptive streaming such as HTTP adaptive streaming (HAS) has become a new trend in multimedia networks [1]. For adapting to network and terminal capabilities, a streaming provider should generate multiple alternatives (or versions) of an original video in advance. Given the current bandwidth, a specific version will be selected for high video quality. HAS can be deployed for constant bitrate (CBR) video or variable bitrate (VBR) video. Basically, VBR videos have larger bitrate variations but more stable visual quality compared to CBR videos.

In HAS, the rate adaptation heuristics are deployed at the client. However, a client-based approach might lead to several negative problems caused by the attempt to optimize the individual quality of each client. The bandwidth competition will occur when the video flows of several clients traverse the same path in the network. This competing behavior among clients can result in incorrect throughput estimations and excessive quality oscillations [2, 3]. A simple solution to overcome these problems could be increasing the capacity of the delivery network but this leads to high costs and complex deployment. Some recent studies [4-8] have proposed network-based solutions to overcome these problems. Nonetheless, the existing methods are limited to CBR videos.

In this paper, we offer the solution to competition problems of streaming multiple VBR videos over a bottleneck where the rate adaptation algorithm can be controlled by components inside the network. With a large number of video streams, an optimal solution for the optimization problem cannot be found easily in real-time. Therefore, we propose an approximation algorithm that can find a nearly optimal solution for the problem with low complexity.

Our goal is to find the allocated bandwidth and the adapted version for each video so that the overall utility is maximized under the limited total available bandwidth and delay. The utility of the video streams is computed using a utility model that takes into account impacts of both video perceptual quality and the end-to-end delay [9]. The experimental results show that the overall utility of the near optimal solution found by the proposed algorithm is very close to that of the optimal solution found by the Full-Search algorithm while the run-time of the proposed algorithm is much smaller.

The rest of the paper is organized as follows. In Section 2, we present a review of related studies. The quality-delay trade-off model is detailed in Section 3. Section 4 provides the formulation and the solution for

* Corresponding author: Tel.: (+84) 988980920
Email: thoa.nguyenthikim@hust.edu.vn

the resource allocation and bitrate adaptation problem when multiple clients share a bottleneck. Finally, the paper is concluded in Section 5.

2. Related work

To improve the users' QoE for HAS services, both client-based, server-based and network-based solutions have been considered. With client-based approach, many rate adaptation heuristics are proposed so far [11-16]. These heuristics are all based on the same principles: servers store multiple versions of an original video as well as related metadata [17]. Based on the information of metadata and status of terminal/networks, the client can decide on which/when media parts are downloaded. In this way, the quality assignment process is performed fully at the client. Consequently, it is difficult to get the fairness when multiple clients sharing a bottleneck. Several algorithms are implemented to tackle aforementioned problem. Villa et al. [18] improve fairness by randomizing the time interval at which client requests a new segment. In this work, they do not make any assessment in term of QoE of users. Thus, they can achieve the optimal network resource utilization, but the quality perceived by the user can be affected. In [3], an ON-OFF pattern is used when the playback buffer size reaches a certain target. Specially, the video player pauses the download when the video buffer is full (OFF period) and the player resumes pulling the data (ON period) when some data in the buffer is consumed. Obviously, this mechanism may not be synchronized among video flows resulting in the inaccurate throughput estimation. Therefore, the bottleneck capacity cannot be fairly shared and the quality adaptation algorithm can fail to optimize QoE of users.

It can be seen that, all presented methods lack the coordination between the clients, so the fairness problem has not been solved thoroughly. To overcome this problem, it is necessary to have a centralized solution. S. Akhsabi et al. [19] propose a server-side traffic shaping approach to minimize oscillations during streaming due to ON-OFF patterns when multiple clients compete for bandwidth. Zhang D et al. [20] propose a server-side-based rate allocation algorithm under Content Delivery Network. They consider user experience in the video bitrate allocation to improve QoE. Although these methods achieve a certain efficiency in the resource allocation and improve QoE, it is difficult, however, to implement the server-based approach in a large scale system.

For large scale system, recent studies [21-22] have introduced network-based solutions to improve fairness and QoE. S.Petrangeli et al. [21] improve fairness by placing intermediary nodes in the network in charge of fair resource sharing among clients. However, this solution just focus on streaming CBR content. Fairness problem in multi-user VBR video streaming is concerned first by Y. Huang et al. [22]. They offer the power allocation for VBR video streaming over multi-cell wireless networks by maximizing the delivered video data under peak transmit power constraint and playout buffer requirements. Though their solution provides a good trade-of between power consumption and buffer utilization, it is mainly based on power management without optimizing channel utilization which may be inefficient in systems limited by spectrum scarcity.

In this paper, we propose a quality-delay trade-off model and apply it in a multiple VBR streams sharing a bottleneck scenario. Our approach achieve the optimal not only in terms of quality adaptation, but also in terms of efficient resource allocation. Furthermore, for each client, the proposed method does not simply select a VBR version but can decide an adapted version to give the user the best possible utility. It should be noted that, CBR videos can be considered as a special case of VBR videos. So the proposed method is still effective when some clients download CBR videos rather than VBR videos.

3. Proposed method

3.1. Problem Formulation

Let us consider a multiple videos streaming system architecture as shown in Fig. 1 where many VBR videos are stored in servers. Information of each video (e.g. adapted bitrate, initial delay, utility corresponding to each level of allocated bandwidth that video can be played) is contained in metadata. Multiple clients of a certain place (e.g. a campus, a building) access the videos via an access link (i.e. the bottleneck). A manager requests metadata from servers and decides the adapted version for each client so that the overall utility of all clients is maximized while meeting the constraints of total bandwidth and delay.

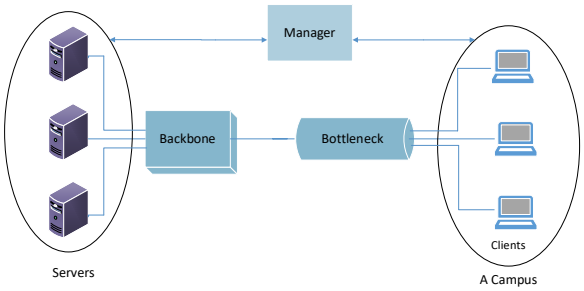

Fig. 1. Multiple videos streaming system architecture.

Assume that the system is simultaneously streaming H videos to the clients. Each video V_i $(1 \leq i \leq H)$ is encoded into M_i versions (with different quality levels), each of which has N_i

segments. Here, the versions of a video are arranged in descending order of the bitrate. The total available bandwidth of the bottleneck is denoted by R^c. The goal of the rate adaptation algorithm at the manager is to determine the version of each video to be selected to maximize the overall utility of the system.

In [23], a quality-delay trade-off model has determined the best adapted version having the highest utility value when streaming a single video with an allocated bandwidth and an initial delay constraint. In this section, we apply this model in the context of multiple streams sharing a bottleneck. Assuming that each video V_i could be allocated K_i different bandwidth levels. The bandwidth at level k_i is denoted by $r_i(k_i)$, $k_i \in [1; K_i]$. Note that $r_i(k_i)$ is less than r_i^{max} that is the required bandwidth to play V_i at the highest quality version with zero initial delay. At each allocated bandwidth level $r_i(k_i)$, we obtain the best adapted content $V_i^*(k_i)$ corresponding to bitrate threshold $\varepsilon_i^*(k_i)$, which having the highest value of utility $u_i^*(k_i)$, the quality $Q_i^*(k_i)$ and the initial delay $d0^*(k_i)$. The adaptation for all video streams can be formulated as an optimization problem as follows.

Find $\{V_i^*(k_i)\}$ for all videos V_i $(k_i \in [1; K_i], i \in [1; H]$ so as to maximize the overall utility U of all video streams

$$U = \sum_{i=1}^{H} w_i \times u_i^*(k_i), i \in [1; H], k_i \in [1; K_i] \quad (1)$$

subject to

$$\sum_{i=1}^{H} r_i(k_i) \le R^c, i \in [1; H], k_i \in [1; K_i] \quad (2)$$

and

$$d0^*(k_i) \le D^c, i \in [1; H], k_i \in [1; K_i] \quad (3)$$

Here, w_i is the weight of the video V_i. This value indicates the importance of content from that video; D^c is the delay constraint of the system.

It can be seen that this optimization problem can be reduced to the 0-1 Knapsack problem and therefore is a NP-hard optimization problem for which an optimal solution cannot be found in real-time [24].

3.2. Optimization Solution

The challenge in this overall utility maximization problem is to determine how much bandwidth to be allocated to each video and which adapted version of each video to be served, by jointly accounting for the amount of available resource and initial delay constraint. In other words, when the resource and initial delay are limited, the manager must determine which adapted version of each video to be streamed so that the total utility of the users can be maximized. In order to solve the aforementioned problem, the general procedure of our solution consists of two main tasks, namely offline processing and online optimization which are presented in detail as follows.

3.2.1 Offline processing

In this task, the model presented in Section 3 is implemented with all different allocated bandwidth levels for each video. As mentioned before, $\{V_i^*(k_i)$, the best adapted version of the video V_i at bandwidth level k_i, is characterized by the following information: allocated bandwidth $r_i(k_i)$, bitrate threshold $\varepsilon_i^*(k_i)$, utility $u_i^*(k_i)$, quality $Q_i^*(k_i)$ and initial delay $d0^*(k_i)$. Therefore, we create a database containing these information of all the best adapted versions corresponding to the different bandwidth levels for all videos. This database is stored as metadata of videos and provided to the manager.

3.2.1 Online processing

The above formulated optimization problem can be optimally solved by the Full-Search algorithm. However, in our scenario, the number of streams or clients could be large. So, in the online proccessing step, a fast approximation algorithm is used for practical applications. Based on the metadata of all videos, we find the adapted version as well as allocated bandwidth for each video to maximize the overall utility in (1) subject to (2) and (3). The proposed algorithm is described as follows.

Assume that each video is originally allocated a minimum bandwidth to play the lowest quality version so that the initial delay is lower than the delay constraint D^c. Let $L[i], (1 \le i \le H)$ be the utility curve for the video V_i which includes k_i points corresponding to the best adapted versions at allocated bandwidth levels. Each point consists of two components: utility (u) and bandwidth cost (r). The first point in $L[i]$ corresponds to the version with the lowest bandwidth cost, and the last point in $L[i]$ corresponds to the version with the highest bandwidth cost. Thus $L[i]$ is a sorted list in the order of increasing bandwidth cost (and also in the order of increasing utility).

Depending on the $(\Delta u/\Delta r)$ ratio, we iteratively improve the overall utility until no more improvement can be obtained. At each iteration, among all videos, we find the one of which $(\Delta u/\Delta r)$ ratio is maximal to improve its utility if its allocated bandwidth is less than or equal to the bandwidth remainder. Also, to reduce computational complexity, firstly, we reduce the number of points in the utility curve by building the convex utility curve $L'[i](1 \le i \le H)$ for each video (Fig. 2). Then, an algorithm which is based on the heapsort algorithm [25] is implemented to quickly find the video which has the best benefit of improvement.

Details of the algorithm are described in Algorithm 1. Finally, the optimal version of each video is identified. Some notations used in this part are clarified in Table 1.

Table 1. Symbols used in the paper

Symbol	Description
BW	The used total bandwidth.
R^c	The total bandwidth of the bottleneck.
$L[i].length$	The number of points in $L[i]$. Here, $L[i].length = K_i$.
$L'[i].length$	The number of points in $L'[i]$.
$L'[i][j]$	Video i at j quality level.
$L'[i][j].u$	The utility of video i at j quality level.
$L'[i][j].r$	The used bandwidth of video i at j quality level.
$L'[i][j].\alpha$	The ratio of improved utility and used bandwidth of video i at j quality level, $L'[i][j].\alpha = \frac{(L'[i][j].u - L'[i][j-1].u)}{(L'[i][j].r - L'[i][j-1].r)}$.
$Reduce(L[i]$	Create the convex curve for $L[i]$.
$push()$	Push an element in to a heap and then re-sort it.
$pop()$	Pop an element from a heap and then re-sort it.
$isEmpty()$	Check a heap, return $true$ value if it is empty and reverse.

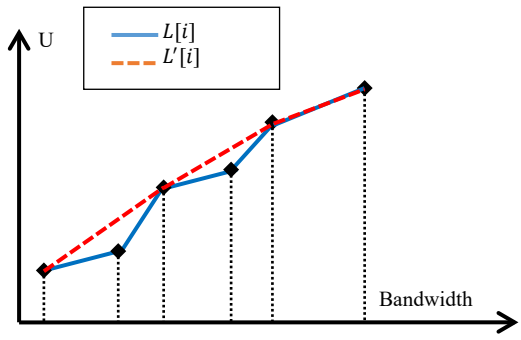

Fig. 2. The utility curves befor and after the reduction.

Algorithm1. Find the optimal version for each video

Input $L[i][j], w_i, 1 \le i \le H, 1 \le j \le K_i$

Output $index[i], 1 \le i \le H$

1: for $1 \le i \le H$ do
2: $L'[i] = Reduce(L[i])$;
3: end for;
4: $BW = 0$;
5: for $1 \le i \le H$ do
6: $index[i] = 0$;
7: $BW \mathrel{+}= L[i][0].bw$;
8: end for;
9: $heap = 0$;
10: for $1 \le i \le H$ do
11: $Node.c = i$;
12: $Node.q = 1$;
13: $Node.\alpha = w_i \times \frac{(L[i][1].u - L[i][0].u)}{(L[i][1].r - L[i][0].r)}$;
14: $heap.push(node)$;
15: end for;
16: while $bw \le R^c$ do
17: $cSelected = -1$;
18: $h = 0$;
19: while ! $heap.isEmpty()$ do
20: $h = heap.pop()$;
21: if $(BW + L[h.c][h.q].r - L[h.c][h.q-1].r \le R^c$ then
22: $cSelected = h.c$;
23: $h.q = h.q + 1$;
24: if $h.q < K_{h.c}$ then
25: $h.\alpha = w_{h.c} \times \frac{(L[h.c][h.q].u - L[h.c][h.q-1].u)}{(L[h.c][h.q].r - L[h.c][h.q-1].r)}$;
26: $heap.push(h)$;
27: end if;
28: if $iSelected \ge 0$ then
29: $index[cSelected] \mathrel{+}= 1$;
30: $BW \mathrel{+}= L[cSelected][index[cSelected]].bw - L[cSelected][index[cSelected - 1]].bw$
31: else
32: $break$;
33: end if;
34: end if;
35: end while;
36: end while;

Let $L_{max} = \max(L[i].length)$ $(1 \leq i \leq H)$, $L'_{max} = \max(L'[i].length)$ $(1 \leq i \leq H)$. In the worst case, the computational complexity of this algorithm is $O(L'_{max} \times H \times logH)$ and the one in building the convex utility curves of all videos is $O(H \times L_{max} \times logL_{max})$. Therefore, the total computational complexity of our algorithm is $O(H \times L_{max} \times logL_{max} + L'_{max} \times H \times logH)$.

3. Experiments and Evaluation

In the first part, we compare the proposed method to the conventional method (called CONV method) where the client selects a version based on the specified bitrate without replacing any video segment. That mean the quality-delay tradeoff model is not applied in the CONV method. The number of different bandwidth levels allocated for each video is set to 10. The delay constrains D^c is set to 0.5 second.

Firstly, we consider the overall utility of both methods. We use 5 videos from the trace in [26]: Silence of the Lambs, Sony Demo, Terminator, Tokyo Olympics and Star Wars IV. These videos are encoded in VBR mode at 6 different QP values which are 22, 28, 34, 38, 42 and 48.

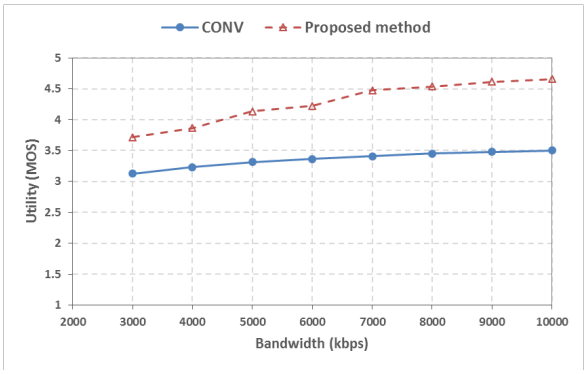

Fig. 3. The utility comparison of the proposed method and the CONV method in streaming the 5 videos.

Fig. 3 shows the utility of the two methods in streaming the 5 videos when the available bandwidth is from 3000kbps to 10000kbps. This figure point out that our proposed significantly improves the utility comparing to the CONV method. It proves that the proposed quality- delay trade-off solution is not only suitable for streaming single video but also suitable for streaming multiple videos.

Secondly, we investigate both the optimality and the run-time of the proposed algorithm and compare it with the Full-Search algorithm. The algorithms have been implemented in C++ and the run-time is measured on an Window 8.1 notebook with an Intel i5-1.7GHz CPU and 6GB memory. The number of streams (H) is changed from 5 to 15 and selected randomly from the 5 videos. We assume that the available throughput is $R^c = 800 \times H$ $(kbps)$ and the weight of each video stream is 1.

The adaptation results are provided in Table 2. It is clearly that the total bandwidth consumption as well as the overall utility are almost the same for both algorithms. The run-times of two methods are showed in Fig. 4. The run-time of the proposed algorithm is negligible, meanwhile that of the Full-Search algorithm increases rapidly as the number of videos increases. With 15 streams, the run-time of the proposed algorithm is less than 0.1 millisecond, while that of the Full-Search algorithm is $17176ms$, corresponding to 17.176 seconds. Thus, it is not acceptable to ensure the real-time of the system. That means the Full-Search algorithm is suitable only for a small-scale network.

Table 2. Bandwidth usage and overall utility of the two algorithms

Number of videos	Full-Search Algorithm		Proposed Algorithm	
	Utility	*Bandwidth (kbps)*	*Utility*	*Bandwidth (kbps)*
5	3.96	3929.4	3.86	3839.8
7	3.44	5482.0	3.44	5482.0
9	3.39	7092.0	3.39	7092.0
11	3.62	8632.4	3.60	8739.4
13	3.39	10327.3	3.39	10327.3
15	3.54	11984.8	3.53	11895.3

Fig. 4 shows the run-time of the proposed method with different numbers of videos. It can be seen that with the proposed algorithm, the run-time is in milliseconds when the number of videos increases to thousands. Thus, the proposed algorithm can be used for large-scale networks.

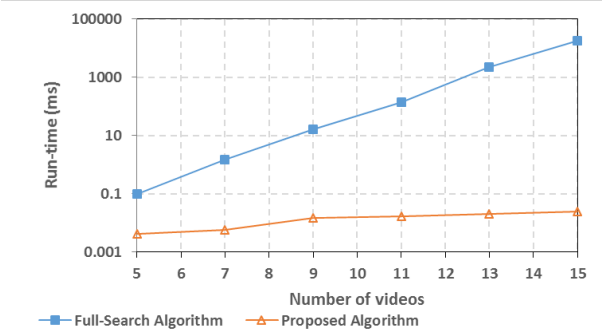

Fig. 4. The run-time in millisecond of the proposed algorithm and Full-Search algorithm.

Table 3. Alocated bandwidth and selected version for each video in the proposed method

R^c (kbps)	Video 1		Video 2		Video 3		Video 4		Video 5	
	R (kbps)	QP	R (kbps)	QP	R (kbps)	QP	R (kbps)	QP	R (kbps)	QP
3000	147	33	582	42	719	40	495	28	478	24
4000	577	24	582	42	719	40	1286	23	674	23
5000	577	24	1857	31	719	40	891	25	871	23
6000	1007	23	1857	31	719	40	1286	23	1067	22
7000	577	24	1857	31	2310	30	1286	23	871	23
8000	1007	23	1857	31	2310	29	1682	23	1067	22
9000	1007	23	1857	31	3902	26	1286	23	871	23
10000	1436	22	1857	31	3902	26	1682	23	1067	22

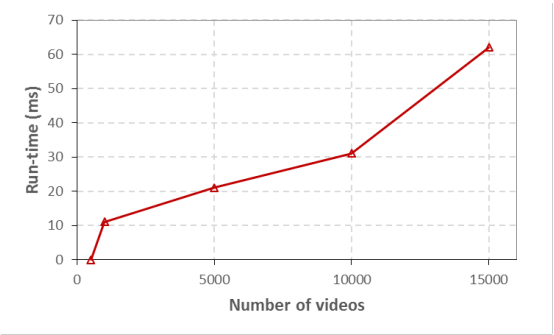

Fig. 5. Run-time of the proposed algorithm.

In the second part, we evaluate the bandwidth alocation and quality adaptation of the proposed method. We use 5 VBR videos [26]: Silence of the Lambs (Video 1), Sony Demo (Video 2), Terminator (Video 3), Tokyo Olympics (Video 4) and Star Wars IV (Video 5). They are encoded at 6 different QP values which are 22, 28, 34, 38, 42 and 48. The total bandwidth R^c of the bottleneck is in the range from 3000kbps to 10000kbps.

Table 3 shows the bandwidth alocation and quality adaptation of the proposed method. Obviously, given an available bandwidth of the bottleneck, the proposed method always determines the alocated bandwidth (R) and selected quality level (QP) for each video, while ensuring that the initial delay is less than D^c and the used total bandwidth ($\sum R$) is less than R^c.

4. Conclusion

In this paper, we have applied the quality-delay trade-off solution to find the best adapted version when streaming single video. After that, we have studied the allocation and adaptation optimization when multiple clients share a bottleneck. We performed an offline process to find all the best adapted versions corresponding to the different bandwidth levels for each VBR video. Then, we proposed an online algorithm for optimizing the bitrate adaptation and bandwidth allocation for multiple clients while achieving the maximal overall utility and meeting the constraints of total bandwidth and delay. The experimental results have shown that the proposed method significantly improves the utility when comparing to the CONV method. The experimental results have also shown that for a large scale system, the proposed algorithm has much better performance in comparison with the Viterbi algorithm in terms of run-time.

Acknowledgments

This research is funded by the Hanoi University of Science and Technology (HUST) under project number T2017-PC-119.

References

[1] T. C. Thang, Hung T. Le, Anh T. Pham and Y. M. Ro, An Evaluation of Bitrate Adaption Methods for HTTP Live Streaming, IEEE J. Selected Areas in Comm., vol.32, no.4, pp.693-705, Apr. 2014.

[2] S. Akhshabi, A. C. Begen, and C. Dovrolis, An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP, in Proc. 2nd Annu. ACM Conf. Multimedia Syst., pp. 157–168, 2011.

[3] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, What happens when HTTP adaptive streaming players compete for bandwidth?, in Proc. Int. Workshop Netw. Oper. Syst. Support Digit. Audio Video, pp. 9–14, 2012.

[4] C. Ben Ameur, E. Mory and B. Cousin, Evaluation of Gateway-Based Shaping Methods for HTTP Adaptive Streaming, in Proc. IEEE ICC Workshop Quality ExperienceBased Manage. Future Internet Applicat. and Services, London, 2015.

[5] A. Essaili et al., QoE-based Traffic and Resource Management for Adaptive HTTP Video Delivery in

LTE, IEEE Trans. Circuits Syst. Video Technol., vol. 25, no. 6, pp. 988-1001, Nov. 2014.

[6] P. Georgopoulos et al., Towards network-wide QoE fairness using openflow-assisted adaptive video streaming, in Proc. ACM SIGCOMM Workshop Future HumanCentric Multimedia Netw., pp. 15–20, 2013.

[7] N. Bouten et al., In-network….QoE optimization through in-network quality adaptation for HTTP adaptive streaming, in Proc. 8th Int. CNSM, Las Vegas, USA, pp. 336–342, 2012.

[8] W. Pu, Z. Zou, and C. W. Chen, Video adaptation proxy for wireless dynamic adaptive streaming over HTTP, in Proc. Int. Packet Video Workshop, pp. 65–70, 2012.

[9] Duc V. Nguyen, Nam Pham Ngoc, Dung M. Nguyen, Anh T. Pham, T. C. Thang, Adaptive home surveillance system using HTTP streaming, in Proc. iCAST-UMEDIA, pp.579-584, Nov. 2013.

[10] G. D. Forney, The Viterbi algorithm, in Proc. IEEE, vol. 61, no. 3, pp. 268-78, Mar. 1973.

[11] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz, Adaptation algorithm for adaptive streaming over http, in Packet Video Workshop (PV), 2012 19th International, May 2012, pp. 173–178.

[12] T.-Y. Huang, R. Johari, N. McKeown, M. Trnnnell, and M. Watson, 'A buffer-based approach to rate adaptation: Evidence from a large video streaming service, in Proc. of the ACM SIGCOMM, 2014, pp. 187-198.

[13] C. Muller, S. Lederer, and C. Timmerer, An evaluation of dynamic adaptive streaming over http in vehicular environments, in Proceedings of the 4th Workshop on Mobile Video, ser. MoVid '12. New York, NY, USA: ACM, 2012, pp. 37–42. [Online]. Available: http://doi.acm.org/10.1145/2151677.2151686

[14] T. C. Thang, H. T. Le, A. T. Pham, Y. M. Ro, An Evaluation of Bitrate Adaptation Methods for HTTP Live Streaming, IEEE Journal on Selected Areas in Communications, vol. 32, no. 4, pp. 693-705, April 2014.

[15] S. García, J. Cabrera, N. García, Quality-Control Algorithm for Adaptive Streaming Services Over Wireless Channels, IEEE Journal of Selected Topics in Signal Processing, vol. 9, no. 1, pp. 50-59, Feb. 2015.

[16] T. C. Thang, H. T. Le, H. X. Nguyen, A. T. Pham, J. W. Kang, Y. M. Ro, Adaptive Video Streaming over HTTP with Dynamic Resource Estimation, IEEE Trans. on Consumer Electronics, vol. 58, no. 1, pp. 78-85, Feb. 2012.

[17] T. Stockhammer, Dynamic Adaptive Streaming over HTTP – Standards and Design Principles, in Proc. ACM MMSys, California, pp. 133-143, Feb. 2011.

[18] B. J. Villa, P. E. Heegaard, and A. Instefjord, Improving fairness for adaptive http video streaming, in Information and Communication Technologies, Robert Szabo and Attila Vidacs (Eds.), Lecture Notes in Computer Science, Vol. 7479. Springer, Berlin, , 2012, pp. 183–193.

[19] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen, Server-based traffic shaping for stabilizing oscillating adaptive streaming players, in Proc. 23rd ACM Int. Workshop NOSSDAV, Oslo, Norway, 2013, pp. 19–24.

[20] Zhang D, He H, Li W, Bitrate allocation among multiple video streams to maximize profit in content delivery networks, Personal and Ubiqitous Computing, vol. 20, no. 3, p. 385-396, 2016.

[21] S. Petrangeli, J. Famaey, M. Claeys, S. Latré, and F. De Turck, QoE-driven Rate Adaptation Heuristic for Fair Adaptive Video Streaming, ACM Trans. Multimedia Comput. Commun. Appl., Oct 2015, 12 (2):28:1–28:24, doi:10.1145/2818361. URL http://doi.acm.org/10.1145/2818361.

[22] Y. Huang, and S. Mao, Downlink power control for multi-user VBR video streaming in cellular networks, IEEE Trans. Multimedia, vol. 15, no. 8, pp. 2137-2148, Dec. 2013.

[23] Duc V. Nguyen, Nam Pham Ngoc, Dung M. Nguyen, Anh T. Pham, T. C. Thang, Quality-Delay Tradeoff Optimization in Multi-Bitrate Adaptive Streaming, in Proc. IEEE ICCE, Las Vegas, US, Jan. 2015.

[24] Garey, Michael R., and Johnson, David S., Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, New York, 1979.

[25] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, Introduction to Algorithms,3rd Edition, MIT Press and McGraw-Hill., 2009.

[26] Geert Van der Auwera, Prasanth T. David, Martin Reisslein, Traffic and Quality Characterization of Single-Layer Video Streams Encoded with H.264/MPEG-4 Advanced Video Coding Standard and Scalable Video Coding Extension, IEEE Trans. Broadcast., vol. 54, no. 3, pp. 698-718, Sep. 2008.