OPTIMAL MULTI-CODEC ADAPTIVE BITRATE STREAMING

Yuriy A. Reznik, Xiangbo Li, Karl O. Lillevold, Abhijith Jagannath, and Justin Greer

Brightcove Inc. 1601 Union St., Ste 1601, Seattle, WA

{yreznik,xli,klillevold,ajagannath,jgreer}@brightcove.com

ABSTRACT

We consider adaptive bitrate streaming system serving a population of mobile devices, where first subset of devices can only decode first codec (e.g. H.264), another subset of devices can only decode the second codec (e.g. HEVC), and where the third sub-set of devices can decode both codecs and can also seamlessly switch between them. We focus on a problem of design of encoding profiles, defining pluralities of streams encoded by both codecs, such that overall quality delivered by such system is optimal. We define this problem mathematically, show how it maps to known classes of optimization problems, and identify approach for solving it numerically. Examples of optimal ladders constructed for different models of networks as well as types of content are provided. It is shown that proposed approach leads to significant improvements compared to systems employing ABR ladders constructed separately for each codec.

***Index Terms*—** ABR streaming; video compression; rate-distortion function; non-linear constrained optimization.

1. INTRODUCTION

In Adaptive Bit-Rate (ABR) streaming systems [1, 2, 3], the content is typically encoded at several bitrates, and where each encoded stream incorporates random access points (e.g. I- or IDR-frames), allowing switching between the streams. During the playback, a *streaming client* monitors the rate at which encoded content is arriving. If such rate becomes insufficient for continuous playback, the client switches to a lower bitrate stream. This prevents buffering. On the other hand, if such rate is greater than the bitrate of the current stream, the client may switch to a higher bitrate stream. This ABR switching mechanism is now widely adopted, and is incorporated in most modern streaming protocols, such as HLS [4], MPEG DASH [5], etc.

The composition of characteristics of streams used for ABR streaming, such as their bitrates, resolutions, codec constraints, etc. is commonly called an *encoding profile* or *ladder*. When first ABR streaming systems were deployed, the encoding profiles were very simple: they typically included 28k, 56k, and 128k streams, corresponding to connection speeds achievable by dial-up and ISDN modems. When faster connections become available, the encoding profiles were extended to include a few higher-bitrate streams. Examples of typical for today's practice profiles can be found in Apple HLS deployment guidelines [6].

In recent years, it was also discovered that the performance of ABR streaming systems can be improved by using *ABR profile generators* which create encoding profiles customarily for each content item and/or properties of networks used for delivery. Such approaches have become known as *per-title*, *content-aware encoding*, and *context-aware encoding* techniques [7, 8, 9, 10].

Most existing ABR profile generators are producing ladders using a single video codec for all streams. For example, an ABR profile can be generated for H.264/MPEG-4 AVC [11] video codec, or another ABR profile can be generate for a newer HEVC [12] codec. This way streaming can be deployed using H.264, and when all devices will support HEVC, new HEVC-encoded streams can be generated and deployed. However, in practice, such switches between codecs do not happen over night. Instead, existing ABR streaming systems are gradually evolving to support multiple codecs, enabling older devices to only use H.264 streams, some newer devices to use HEVC streams, and some other newer devices (such as e.g. all recent Apple devices) to use both H.264 and HEVC streams and to switch between them.

Considering such hybrid multi-codec deployments, it becomes clear that ABR profile must be generated differently, considering all codecs that need to be supported, as well as partition of population of receiving devices into categories that can decode different subsets of such codecs. This is precisely the problem that this paper is intended to address. In Section 2, we offer mathematical definition of this problem, considering, for simplicity, a system with 2 codecs and 3 types of receiving devices. In Section 3, we propose a numerical approach for solving this problem. Experimental results, considering specific examples of content and network models are presented in Section 4. Extensions and concluding remarks are offered in Section 5.

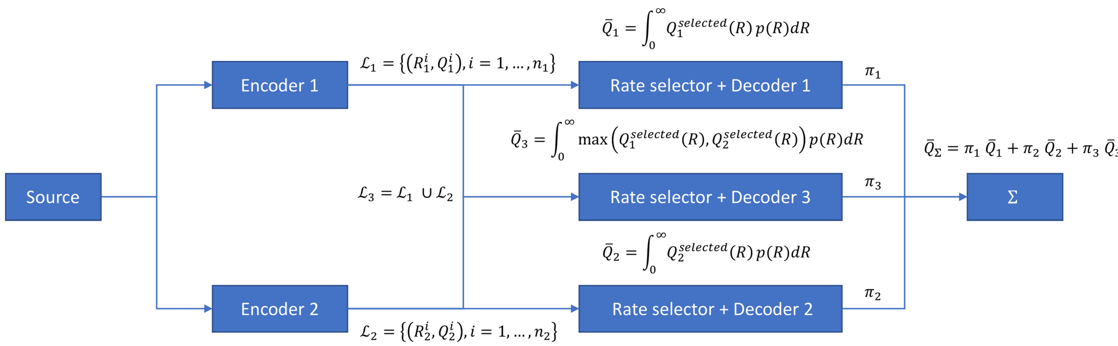

Fig. 1: Model of ABR streaming system with 2 types of codecs and 3 clients

.

2. DEFINITION OF THE PROBLEM

2.1. Encoding ladders and quality-rate functions

In following description, by letter R we denote bitrates, and by letter Q we denote quality values achievable by video codecs. We assume that quality values Q are normalized, such that value $Q = 0$ represents worst possible quality, and $Q = 1$ represents ideal reconstruction.

We next consider 2 codecs, which for a given content, can be used to produce encodings with the following sets of (quality,rate) points: $\mathcal{L}_1 = \left\{\left(Q_1^i, R_1^i\right), i = 1, ..., n_1\right\}$ and $\mathcal{L}_2 = \left\{\left(Q_2^i, R_2^i\right), i = 1, ..., n_2\right\}$. The sub-indices in both cases indicate codec type.

More generally, we also assume that performance of such codecs can be modeled by certain *quality-rate functions*: $Q_1(R)$ and $Q_2(R)$, and that above sets are just sample values from these functions: $Q_1^i = Q_1\left(R_1^i\right), i = 1, ..., n_1$, and $Q_2^i = Q_2\left(R_1^i\right), i = 1, ..., n_2$.

We call the sets $\mathcal{L}_1$ and $\mathcal{L}_2$ *encoding ladders* for codecs of type 1 and 2 respectively. The union of both sets $\mathcal{L} = \mathcal{L}_1 \cup \mathcal{L}_1$ we call a *dual-codec ladder*. For convenience of notation, we also assume that such ladders can always be augmented by zero point: $\left(R^0, Q^0\right) = (0, 0)$, which is same for both codecs.

2.2. Ladder filtering and client models

As common for today's ABR streaming deployments, we assume that the manifest describing the complete dual-codec ladder $\mathcal{L}$ will be filtered on its way to the clients, such that clients that can only support 1st codec will only see $\mathcal{L}_1$ subset and clients that can only support 2nd codec will see $\mathcal{L}_2$. Further, the clients, capable of decoding both codecs would receive the entire set $\mathcal{L}$.

Next, we assume that rate selection logic in clients can be modeled by *conservative selection rule* [9], where given current available bandwidth R clients simply pick the largest rate available in the profile that does not exceed R. For clients capable of decoding codecs 1 or 2 this means, respectively:

$$R_1^{selected}(R) = \max_{\substack{i=0,\dots,n \\ R_1^i \le R}} R_1^i, \quad R_2^{selected}(R) = \max_{\substack{i=0,\dots,n \\ R_2^i \le R}} R_2^i.$$

The resulting quality of streams selected by each client can be expressed as follows:

$$Q_1^{selected}(R) = \max_{\substack{i=0,\dots,n \\ R_1^i \le R}} Q_1^i, \quad Q_2^{selected}(R) = \max_{\substack{i=0,\dots,n \\ R_2^i \le R}} Q_2^i.$$

For clients capable of decoding both codecs, we assume that their selection rule is more intelligent, delivering best quality achievable by using encodings from both sets: $\mathcal{L}_1$ and $\mathcal{L}_2$:

$$R_3^{selected}(R) = \begin{cases} R_1^{selected}(R), & \text{if } Q_1^{selected}(R) \geq Q_2^{selected}(R), \\ R_2^{selected}(R), & \text{otherwise} \end{cases}$$

The resulting quality in dual-codec client is:

$$Q_3^{selected}(R) = \max\left(Q_1^{selected}(R), Q_2^{selected}(R)\right).$$

Such rules are indeed very simplistic, but as explained in [9], they are adequate for studying bandwidth usage of streaming systems in the average case.

2.3. Average quality achievable by each client

Given the rate selection rules, and by assuming that network bandwidth can be modeled as a continuous random variable R with probability density function $p(R)$, we can now produce expressions for *average quality* achievable by clients of each kind:

$$\bar{Q}_1 = \int_0^\infty Q_1^{selected}(R)p(R)dR,$$

$$\bar{Q}_2 = \int_0^\infty Q_2^{selected}(R)p(R)dR,$$

$$\bar{Q}_3 = \int_0^\infty \max\left(Q_1^{selected}(R), Q_2^{selected}(R)\right) p(R)dR.$$

$$\bar{Q}_\Sigma\left(p, \pi, \hat{R}_1^1, ..., \hat{R}_1^{\hat{n}_1}, \hat{R}_2^1, ..., \hat{R}_2^{\hat{n}_2}\right) = \max_{\substack{n_1+n_2=n \\ R_{min} \le R_1^1 \le ... \le R_1^{n_1} \le R_{max} \\ R_{min} \le R_2^1 \le ... \le R_2^{n_2} \le R_{max} \\ R_1^1, R_2^1 < R_max}} \bar{Q}_\Sigma\left(p, \pi, R_1^1, ..., R_1^{n_1}, R_1^1, ..., R_2^{n_2}\right) \quad (1)$$

Next, by assuming that $\pi = \{\pi_1, \pi_2, \pi_3\}$ denotes distribution of clients of each kind in overall population of clients, we can now express *overall average quality across* achievable by the streaming system:

$$\bar{Q}_\Sigma = \pi_1 \bar{Q}_1 + \pi_2 \bar{Q}_2 + \pi_3 \bar{Q}_3.$$

We illustrate the overall flow of the above definitions arriving at final quality expression in Figure 1.

2.4. The ladder design problem

Considering all above plus an observation that average quality value $\bar{Q}_\Sigma$ can be understood as a function of network bandwidth density $p(R)$, client distribution π, and sets of rates used in the ladder, we are now ready to define the lader design problem. Given:

- quality-rate function $Q(R)$,
- network bandwidth density $p(R)$,
- limits for all rate points: R_{min}, R_{max};
- maximum limits for first rate points: $R_{1,max}$,
- the total number of points n, and
- probabilities of usage of clients π,

Find:

- numbers $\hat{n}_1$, $\hat{n}_2$, such that $\hat{n}_1 + \hat{n}_2 = n$, and
- ladder rates $\hat{R}_1^1, ..., \hat{R}_1^{\hat{n}_1}, \hat{R}_2^1, ..., \hat{R}_2^{\hat{n}_2}$

such that overall quality delivered by the system $\bar{Q}_\Sigma$ is maximal. This problem, expressed in mathematical notation is shown in (1).

3. FINDING OPTIMAL MULTI-CODEC LADDERS

3.1. Solving the optimization problem

As easily noticed, the problem (1) is a non-linear constrained optimization problem, where certain complications are added by the fact that $\bar{Q}_\Sigma\left(p, \pi, R_1^1, ..., R_1^{n_1}, R_1^1, ..., R_2^{n_2}\right)$ is not differentiable (due to the use of max operator in quality decision for mixed client), and the fact that the choice of $\hat{n}_1 + \hat{n}_2 = n$ falls in the discrete domain while the rest is continuous.

In order to arrive at practical solution, we first split this problem in continuous and discrete parts, where the latter is concerned with finding best pair of numbers $\hat{n}_1$, and $\hat{n}_2$ such that $\hat{n}_1 + \hat{n}_2 = n$. We implement it by using bruteforce search across all values of $\hat{n}_1 = 0, ..., n$, and then selecting the result that achieves the best quality. To solve the optimization problem for each given pair of numbers $\hat{n}_1$, and $\hat{n}_2$, we use sequential quadratic programming [13].

3.2. Ladder performance parameters

The solutions of the optimization problem (1) are effectively the sets of ladder points $\hat{R}_1^1, ..., \hat{R}_1^{\hat{n}_1}$, and $\hat{R}_2^1, ..., \hat{R}_2^{\hat{n}_2}$, and the associated average quality values $\bar{Q}_i, i = 1, .., 3$ and $\bar{Q}_\Sigma$.

In order to understand the effectiveness of such ladder in relative sense, we will also use the *quality gap* parameters:

$$\xi_i = \frac{Q_i^* - \bar{Q}_i}{Q_i^*}, i = 1, .., 3, \quad \xi_\Sigma = \frac{Q_\Sigma^* - \bar{Q}_\Sigma}{Q_\Sigma^*},$$

where:

$$Q_1^* = \int_0^\infty Q_1(R) p(R) dR, \;\; Q_2^* = \int_0^\infty Q_2(R) p(R) dR,$$

$$Q_3^* = \int_0^\infty \max\left(Q_1(R), Q_2(R)\right) p(R) dR,$$

$$Q_\Sigma^* = \pi_1 Q_1^* + \pi_2 Q_2^* + \pi_3 Q_3^*.$$

are effectively *quality limits* achievable when the number of ladder points approaches infinity ($n \to \infty$):

In other words, parameters $\xi_i, i = 1, .., 3$ and ξ_Σ indicate how well a ladder with given n points performs relative to quality achievable for given content, codecs, clients, and networks in principle.

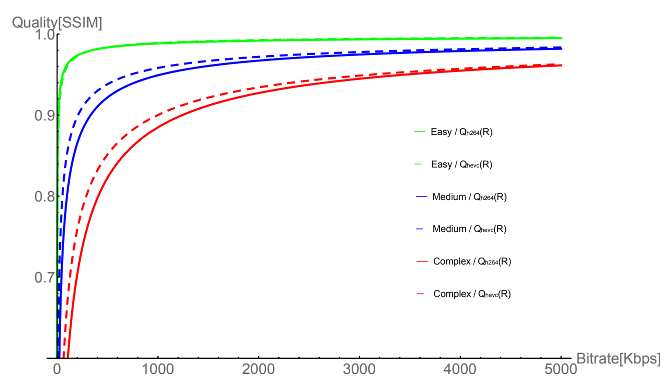

Fig. 2: Quality-rate models constructed for h264 and HEVC encoders for 3 types of content.

4. EXPERIMENTAL RESULTS

4.1. Testing setup

For the purpose of our experiments we use 3 video sequences, which we call "Easy", "Medium", and "Complex" describing the degrees of challenge that they present to the encoder. They were produced by catenating several standard test sequences described in [14]. Specific composition of sequences in each test are shown in Table 1.

Table 1: Test Sequences.

Test sequence	Resolution	Component sequences
Easy	720p	Johnny KristenAndSara FourPeople
Medium	720p	ParkScene FourPeople BasketballDrive Traffic
Complex	720p	BQTerrace BasketballDrive Cactus PeopleOnStreet NebutaFestival

As encoders we use open source x264 [15] and x265 [16] encoders, implementing H.264/MPEG-4 AVC [11] and HEVC [12] standards respectively. For all encodings we employ typical constraints as used for ABR streaming in practice (main profile, ABR rate control, 2sec GOP, $1.5\times$ ratio of maximum bitrate to target bitrate, and $2.0\times$ ratio of decoder buffer size to target bitrate, etc). For measuring quality we use SSIM quality metric [17]. The following model function was employed to fit the (quality,rate) points produced as a result of encodings:

$$Q_{\alpha,\beta}(R) = \frac{R^\beta}{\alpha^\beta + R^\beta} \tag{2}$$

In Table 2, we show the values of model parameters α, and β obtained for our codecs and content, and in Figure 2 we show shapes of the obtained quality-rate functions.

Table 2: Parameters of quality-rate models.

Test sequence	H.264		HEVC	
	α	β	α	β
Easy	0.1935	0.5600	0.3645	0.5674
Medium	12.0449	0.6623	5.1552	0.5947
Complex	60.9995	0.7295	34.7613	0.6548

To obtain network bandwidth models, we used throughput measurements of LTE network [18], fitted to the following model functions:

$$p_{\alpha,\sigma_1,\sigma_2}(R) = \alpha\, f\,(R|\sigma_1) + (1-\alpha) f\,(R|\sigma_2)$$

where,

$$f\,(R|\sigma) = \frac{x}{\sigma}\, e^{-\frac{x^2}{2\sigma^2}}$$

is the probability density function of Rayleigh distribution, α, σ_1, and σ_2 are model parameters. As shown Table 3 and Figure 3 (right), two models are obtained by scaling network throughput by two possible numbers of users in the LTE cell.

Table 3: Parameters of network models.

Network	Model Parameters		
	a	σ_1	σ_2
Network 1	0.4287	901.10	2249.64
Network 2	0.4287	1802.20	4499.27

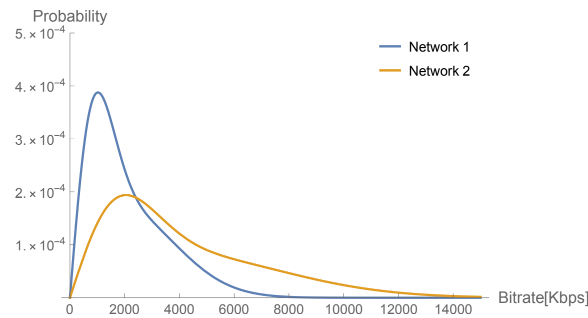

Fig. 3: Network models used in our experiments

For rate constraints we use: $r_{\min} = 50$, $r1_{\max} = 500$, and $r_{\max} = 10000$ [Kbps]. We further assume that overall population of clients consists of: 60% of devices that can only decode H.264, 30% of devices that can decode and switch between H.264 and HEVC streams (e.g. Apple devices), and 10% of devices that can decode HEVC but cannot switch to H.264 (e.g. Android/DASH devices).

4.2. The results

The optimal ladders constructed for $n = 2, ..., 8$, three types of content, and two network models are shown in Tables 4 and 5 respectively. These table include descriptions of rates and codecs selected for each encoded point, as well as best and average quality values achievable by clients of each kind. E.g. for the first, H.264-only codec client, we show the number of ladder points used n_1, quality achieved at top rendition $Q_1^{n_1}$, average quality $\bar{Q}_1$, and quality gap ξ_1. The same is shown for 2 other types of clients, and finally we also show weighted average quality $\bar{Q}_\Sigma$ and quality gap ξ_Σ considering all clients in the system.

4.3. Observations

Based on the above results, several observations can be made.

First, the proposed optimization framework has shown to be operational, producing ladders with progressively improving characteristics with increasing number of rate points n.

Second, the produced ladders look different for different content: the "Easy" content gets encoded with fewer bits and higher quality values achieved, while profiles generated for "Difficult" content use more bits and their quality values are lower.

Table 4: Quality-optimal ladders generated for Network Model 1.

Content	n	Ladder bitrates & codecs: H.264 and HEVC	Results for H.264-only clients				Results for HEVC-only clients				Results for dual-codec clients				Weighted average across all clients	
			n_1	$Q_1^{n_1}$	$\overline{Q}_1$	ξ_1	n_2	$Q_2^{n_2}$	$\overline{Q}_2$	ξ_2	n_3	$Q_3^{n_3}$	$\overline{Q}_3$	ξ_3	$\overline{Q}_\Sigma$	ξ_Σ
Easy	2	156(h264), 164(hevc)	1	0.9701	0.9626	2.78	1	0.9697	0.9614	2.94	1	0.9701	0.9626	2.78	0.9625	2.79
	3	66(h264), 636(h264), 164(hevc)	2	0.9854	0.9804	0.98	1	0.9697	0.9614	2.94	3	0.9854	0.9821	0.81	0.9790	1.12
	4	50(h264), 366(h264), 1155(h264), 164(hevc)	3	0.9892	0.9844	0.57	1	0.9697	0.9614	2.94	4	0.9892	0.9851	0.50	0.9823	0.79
	5	50(h264), 366(h264), 1155(h264), 70(hevc), 633(hevc)	3	0.9892	0.9844	0.57	2	0.9857	0.9803	1.04	5	0.9892	0.9856	0.46	0.9843	0.58
	6	50(h264), 280(h264), 744(h264), 1680(h264), 70(hevc), 633(hevc)	4	0.9911	0.9860	0.41	2	0.9857	0.9803	1.04	6	0.9911	0.9864	0.37	0.9855	0.46
	7	50(h264), 232(h264), 562(h264), 1087(h264), 2153(h264), 70(hevc), 633(hevc)	5	0.9922	0.9868	0.33	2	0.9857	0.9803	1.04	7	0.9922	0.9871	0.30	0.9863	0.39
	8	50(h264), 232(h264), 562(h264), 1087(h264), 2153(h264), 50(hevc), 355(hevc), 1126(hevc)	5	0.9922	0.9868	0.33	3	0.9896	0.9846	0.60	8	0.9922	0.9873	0.28	0.9867	0.34
Medium	2	327(h264), 283(hevc)	1	0.8990	0.8691	9.21	1	0.9154	0.8924	7.49	1	0.9154	0.8924	7.49	0.8784	8.52
	3	167(h264), 836(h264), 283(hevc)	2	0.9431	0.9182	4.08	1	0.9154	0.8924	7.49	3	0.9431	0.9287	3.73	0.9188	4.31
	4	114(h264), 489(h264), 1304(h264), 283(hevc)	3	0.9570	0.9328	2.56	1	0.9154	0.8924	7.49	4	0.9570	0.9375	2.82	0.9301	3.13
	5	88(h264), 348(h264), 815(h264), 1750(h264), 283(hevc)	4	0.9643	0.9396	1.84	1	0.9154	0.8924	7.49	4	0.9643	0.9430	2.24	0.9359	2.53
	6	88(h264), 348(h264), 815(h264), 1750(h264), 139(hevc), 795(hevc)	4	0.9643	0.9396	1.84	2	0.9524	0.9339	3.19	5	0.9643	0.9461	1.92	0.9410	2.00
	7	71(h264), 268(h264), 595(h264), 1108(h264), 2149(h264), 139(hevc), 795(hevc)	5	0.9687	0.9436	1.43	2	0.9524	0.9339	3.19	6	0.9687	0.9482	1.71	0.9440	1.69
	8	71(h264), 268(h264), 595(h264), 1108(h264), 2149(h264), 93(hevc), 459(hevc), 1275(hevc)	5	0.9687	0.9436	1.43	3	0.9636	0.9456	1.97	7	0.9687	0.9511	1.40	0.9460	1.47
Complex	2	469(h264), 417(hevc)	1	0.8157	0.7614	15.8	1	0.8358	0.7912	13.7	1	0.8358	0.7912	13.7	0.7734	15.0
	3	265(h264), 1009(h264), 417(hevc)	2	0.8856	0.8334	7.89	1	0.8358	0.7912	13.7	3	0.8856	0.8517	7.08	0.8346	8.23
	4	190(h264), 625(h264), 1496(h264), 417(hevc)	3	0.9117	0.8579	5.18	1	0.8358	0.7912	13.7	4	0.9117	0.8664	5.48	0.8538	6.13
	5	150(h264), 460(h264), 959(h264), 1950(h264), 417(hevc)	4	0.9260	0.8703	3.81	1	0.8358	0.7912	13.7	4	0.9260	0.8760	4.43	0.8641	4.99
	6	150(h264), 460(h264), 959(h264), 1950(h264), 228(hevc), 960(hevc)	4	0.9260	0.8703	3.81	2	0.8978	0.8559	6.62	6	0.9260	0.8811	3.87	0.8721	4.11
	7	124(h264), 364(h264), 715(h264), 1246(h264), 2322(h264), 228(hevc), 960(hevc)	5	0.9343	0.8776	3.00	2	0.8978	0.8559	6.62	7	0.9343	0.8856	3.38	0.8779	3.48
	8	106(h264), 301(h264), 571(h264), 940(h264), 1497(h264), 2609(h264), 228(hevc), 960(hevc)	6	0.9393	0.8824	2.48	2	0.8978	0.8559	6.62	7	0.9393	0.8888	3.04	0.8817	3.06

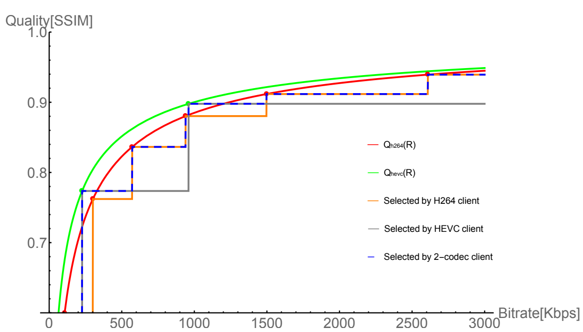

Fig. 4: 7-point profile constructed for Complex content and Network 1, and quality levels achievable by different clients.

Third, produced ladders are also different for different networks. For profiles produced for Network 1 we observe concentration of rate points around 1000kbps. For Network 2, the concentration of points shifts closer to 2000 kbps. Such concentration points represent peaks of bandwidth distributions in each network model.

Finally, we also observe that the distribution of rates and points between H.264 and HEVC codecs in optimal profiles *follow a certain interleaved pattern*, allowing dual-codec client to alternate between these codecs and achieve more fine-grain adaptation and better quality as a result. Specifically, as shown in in Figure 4, it can be seen how a profile including 5 H.264 streams and just 2 HEVC streams can be used by dual-codec client to perform adaptation, achieving better quality than achievable by using only H.264 or HEVC streams.

Figure 4 also shows that jointly generated profiles are much more efficient in terms of total number of required streams. In this case, the hybrid profile uses only 2 HEVC streams, and they are sufficient to increase overall quality. The approach relying on independent profile generation would have required 5 HEVC streams (same as number of H.264 streams) to enable similar deployment.

5. EXTENSIONS AND CONCLUDING REMARKS

The proposed framework for analysis and design of optimal multi-codec ABR profiles can be altered or extended in variety of ways.

To account for multiple possible resolutions $s \in S$ it is sufficient to obtain resolution-specific quality-rate functions $Q_i(R, s)$, and then take upper boundary $Q_i(R) = \sup_{s \in S} Q_i(R, s)$ as final quality-rate function to be used in the optimization process.

The addition of extra codecs and categories of receiving devices can be handled by extra quality-rate models $Q_i(R)$, as well as client rate selection models $R_j^{selected}(R)$, considering permutations of supported codecs and switching capabilities of each such client.

Finally, as also mentioned in [9], the client models may also be modified to allow more aggressive behavior, and used e.g. to study stability and optimality of the streaming system considering variations of clients.

Table 5: Quality-optimal ladders generated for Network Model 2.

Content	n	Ladder bitrates & codecs: H.264 and HEVC	Results for H.264-only clients				Results for HEVC-only clients				Results for dual-codec clients				Weighted average across all clients	
			n_1	$Q_1^{n_1}$	$\overline{Q}_1$	ξ_1	n_2	$Q_2^{n_2}$	$\overline{Q}_2$	ξ_2	n_3	$Q_3^{n_3}$	$\overline{Q}_3$	ξ_3	$\overline{Q}_\Sigma$	ξ_Σ
Easy	2	271(h264), 282(hevc)	1	0.9774	0.9717	2.15	1	0.9776	0.9714	2.24	2	0.9776	0.9718	2.14	0.9717	2.16
	3	110(h264), 1206(h264), 282(hevc)	2	0.9895	0.9859	0.72	1	0.9776	0.9714	2.24	3	0.9895	0.9872	0.59	0.9848	0.84
	4	68(h264), 636(h264), 2172(h264), 282(hevc)	3	0.9922	0.9890	0.41	1	0.9776	0.9714	2.24	4	0.9922	0.9895	0.36	0.9874	0.58
	5	68(h264), 636(h264), 2172(h264), 116(hevc), 1192(hevc)	3	0.9922	0.9890	0.41	2	0.9900	0.9862	0.75	5	0.9922	0.9900	0.30	0.9890	0.41
	6	51(h264), 429(h264), 1309(h264), 3108(h264), 116(hevc), 1192(hevc)	4	0.9935	0.9902	0.28	2	0.9900	0.9862	0.75	5	0.9935	0.9906	0.25	0.9899	0.32
	7	50(h264), 347(h264), 968(h264), 1992(h264), 4060(h264), 116(hevc), 1192(hevc)	5	0.9944	0.9909	0.22	2	0.9900	0.9862	0.75	7	0.9944	0.9913	0.18	0.9906	0.26
	8	50(h264), 347(h264), 968(h264), 1992(h264), 4060(h264), 71(hevc), 624(hevc), 2128(hevc)	5	0.9944	0.9909	0.22	3	0.9928	0.9894	0.43	8	0.9944	0.9915	0.16	0.9909	0.22.
Medium	2	500(h264), 486(hevc)	1	0.9218	0.9037	7.06	1	0.9372	0.9198	5.78	1	0.9372	0.9198	5.78	0.9101	6.55
	3	272(h264), 1536(h264), 486(hevc)	2	0.9612	0.9440	2.92	1	0.9372	0.9198	5.78	3	0.9612	0.9512	2.56	0.9437	3.10
	4	181(h264), 875(h264), 2452(h264), 486(hevc)	3	0.9713	0.9550	1.79	1	0.9372	0.9198	5.78	4	0.9713	0.9581	1.85	0.9524	2.21
	5	181(h264), 875(h264), 2452(h264)), 229(hevc),1480(hevc)	3	0.9713	0.9550	1.79	2	0.9666	0.9534	2.33	5	0.9713	0.9616	1.50	0.9568	1.76
	6	138(h264), 610(h264), 1510(h264), 3318(h264), 229(hevc), 1480(hevc)	4	0.9764	0.9600	1.27	2	0.9666	0.9534	2.33	5	0.9764	0.9638	1.27	0.9605	1.38
	7	112(h264), 465(h264), 1090(h264), 2090(h264), 4107(h264), 229(hevc), 1480(hevc)	5	0.9794	0.9628	0.98	2	0.9666	0.9534	2.33	7	0.9794	0.9657	1.08	0.9628	1.14
	8	93(h264), 372(h264), 842(h264), 1530(h264), 2580(h264), 4676(h264), 229(hevc), 1480(hevc)	6	0.9811	0.9646	0.80	2	0.9666	0.9534	2.33	7	0.9811	0.9667	0.98	0.9641	1.00
Complex	2	500(h264), 500(hevc)	1	0.8227	0.8065	14.1	1	0.8514	0.8346	11.7	1	0.8514	0.8346	11.7	0.8177	13.1
	3	428(h264), 1821(h264), 500(hevc)	2	0.9225	0.8856	5.73	1	0.8514	0.8346	11.7	3	0.9225	0.8946	5.32	0.8832	6.20
	4	300(h264), 1096(h264), 2750(h264), 500(hevc)	3	0.9415	0.9049	3.67	1	0.8514	0.8346	11.7	4	0.9415	0.9111	3.57	0.8997	4.44
	5	300(h264), 1096(h264), 2750(h264), 374(hevc), 1758(hevc)	3	0.9415	0.9049	3.67	2	0.9288	0.8986	4.89	5	0.9415	0.9169	2.95	0.9079	3.58
	6	234(h264), 791(h264), 1737(h264), 3616(h264), 374(hevc), 1758(hevc)	4	0.9516	0.9143	2.67	2	0.9288	0.8986	4.89	6	0.9516	0.9208	2.54	0.9146	2.86
	7	193(h264), 618(h264), 1280(h264), 2302(h264), 4360(h264), 374(hevc), 1758(hevc)	5	0.9575	0.9197	2.09	2	0.9288	0.8986	4.89	7	0.9575	0.9249	2.10	0.9192	2.37
	8	164(h264), 506(h264), 1012(h264), 1721(h264), 2795(h264), 4957(h264), 374(hevc), 1758(hevc)	6	0.9611	0.9233	1.71	2	0.9288	0.8986	4.89	7	0.9611	0.9266	1.93	0.9218	2.09

6. REFERENCES

[1] D. Wu, Y.T. Hou, W. Zhu, Y-Q. Zhang, and J.M. Peha, "Streaming video over the internet: approaches and directions," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 282–300, 2001.

[2] G. J. Conklin, G. S. Greenbaum, K. O. Lillevold, A. F. Lippman, and Y. A. Reznik, "Video coding for streaming media delivery on the internet," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 269–281, March 2001.

[3] B. Girod, M. Kalman, Y.J. Liang, and R. Zhang, "Advances in channel-adaptive video streaming," *Wireless Communications and Mobile Computing*, vol. 2, no. 6, pp. 573–584, 2002.

[4] R. Pantos and W. May, "HTTP live streaming, RFC 8216," `https://tools.ietf.org/html/rfc8216`, August 2017.

[5] ISO/IEC 23009-1:2012, "Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats," February 2012.

[6] Apple Inc., "HLS authoring specification for Apple devices," `https://developer.apple.com/documentation/http_live_streaming/hls_authoring_specification_for_apple_devices`, September 2018.

[7] A. Aaron et al., "Per-title encode optimization," `https://medium.com/netflix-techblog/per-title-encode-optimization-7e99442b62a2`, December 15 2015, Netflix technology blog.

[8] Ultra HD Forum, "Ultra HD Forum phase B guidelines," `https://ultrahdforum.org/wp-content/uploads/Ultra-HD-Forum-Phase-B-Guidelines-v1.0.pdf`, April 2018.

[9] Y. A. Reznik, K. O. Lillevold, A. Jagannath, J. Greer, and J. Corley, "Optimal design of encoding profiles for abr streaming," in *Proceedings of the 23rd Packet Video Workshop*, New York, NY, USA, 2018, PV '18, pp. 43–47, ACM.

[10] C. Chen, Y. Lin, S. Benting, and A. Kokaram, "Optimized transcoding for large scale adaptive streaming using playback statistics," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, Oct 2018, pp. 3269–3273.

[11] ISO/IEC 14496-10:2003, "Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding," December 2003.

[12] ISO/IEC 23008-2:2013, "Information technology – High efficiency coding and media delivery in heterogeneous environments – Part 2: High efficiency video coding," December 2013.

[13] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer-Verlag, New York, NY, USA, 2006.

[14] F. Bossen, "Common test conditions and software reference configurations," JCT-VC output document JCTVC-H1100, February 2012.

[15] "x264 open source encoder project," `https://www.videolan.org/developers/x264.html`.

[16] "x265 open source encoder project," `https://www.x265.org/`.

[17] Z. Wang, L. Lu, and A. C. Bovik, "Video quality assessment based on structural distortion measurement," *Signal Processing: Image Communication*, vol. 19, no. 2, pp. 121 – 132, 2004.

[18] J. Karlsson and M. Riback, "Initial field performance measurements of LTE," *Ericsson review*, vol. 3, 2008.