Optimal Design of Encoding Profiles for ABR Streaming

Yuriy A. Reznik, Karl O. Lillevold, Abhijith Jagannath, Justin Greer, and Jon Corley
Brightcove, Inc., Seattle, WA, USA.

ACM MMSys | Packet Video Workshop, June 12, 2018, Amsterdam, The Netherlands

Some facts from history

1948 – C. Shannon: rate-distortion theory, source & channel coding theorems

1970s – experiments with DCT, first image, video, and audio codecs

1980s – emergence of Internet

…

1995 – RealAudio – first Internet streaming audio system

1997 – RealVideo, SureStream, RealSystem G2 – first ABR streaming system

…

2007 – Move Networks, first HTTP-based ABR streaming

2009 – Apple HLS, Microsoft Smooth, Adobe HDS

2011 – MPEG DASH

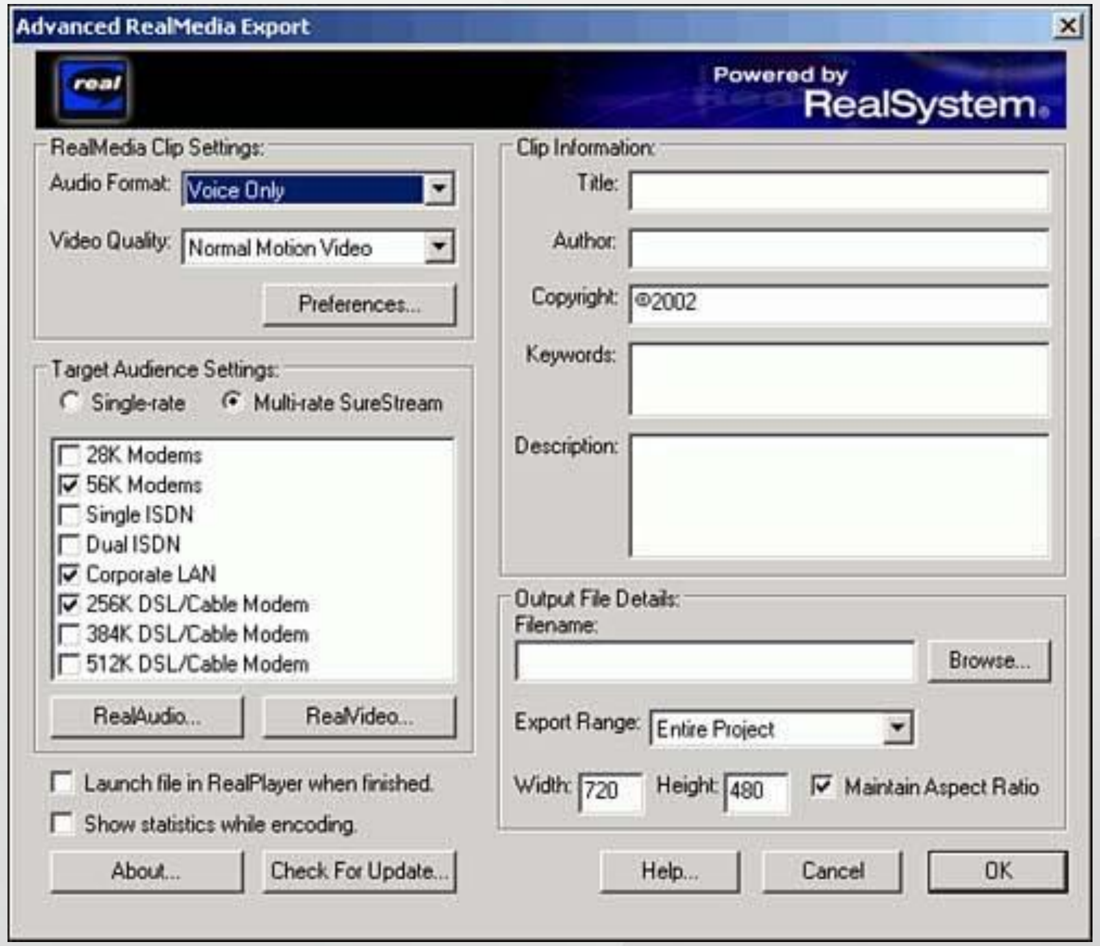
2014 – CMAF

…

2015 – Netflix “Per-title Encoding”: exploiting statistics of the source

2017 – Brightcove “Context Aware Encoding”: exploiting statistics of the networks

…



ABR encoding profile design dialog in RealSystem 8 (2001)

brightcove

Examples of modern-era ABR encoding profiles

Apple HLS authoring specification:

HEVC/H.265	H.264/AVC	Resolution	Frame rate
145	145	416 x 234	≤ 30 fps
350	365	480 x 270	≤ 30 fps
660	730	640 x 360	≤ 30 fps
990	1100	768 x 432	≤ 30 fps
1700	2000	960 x 540	same as source
2400	3000	1280 x 720	same as source
3200	4500	same as source	same as source
4500	6000	same as source	same as source
5800	7800	same as source	same as source

Brightcove “High-Resolution” profile

media type	video codec	video bitrate	decoder bitrate cap	decoder buffer size	max frame rate	width	height	h264 profile
video	h264	450	771	1028	30	480	270	baseline
video	h264	700	1194	1592	30	640	360	baseline
video	h264	900	1494	1992	30	640	360	main
video	h264	1200	1944	2592	30	960	540	main
video	h264	1700	2742	3656	30	960	540	main
video	h264	2500	3942	5256	30	1280	720	main
video	h264	3500	5442	7256	30	1920	1080	high
video	h264	3800	6192	8256	30	1920	1080	high

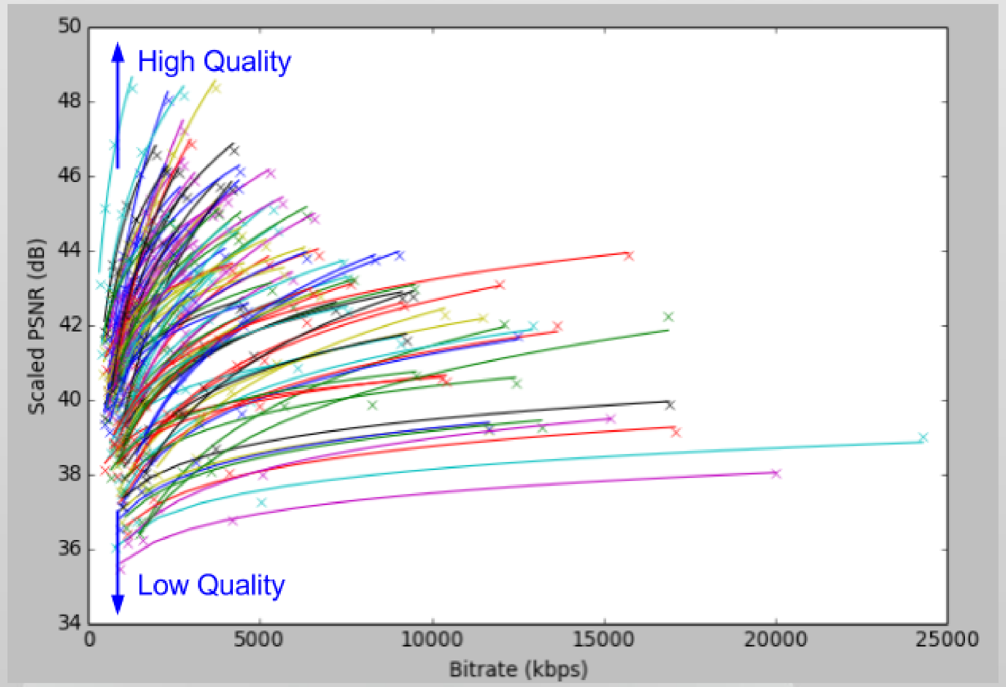
Static vs Dynamic encoding ladders

All previously shown examples are so-called **static** encoding ladders

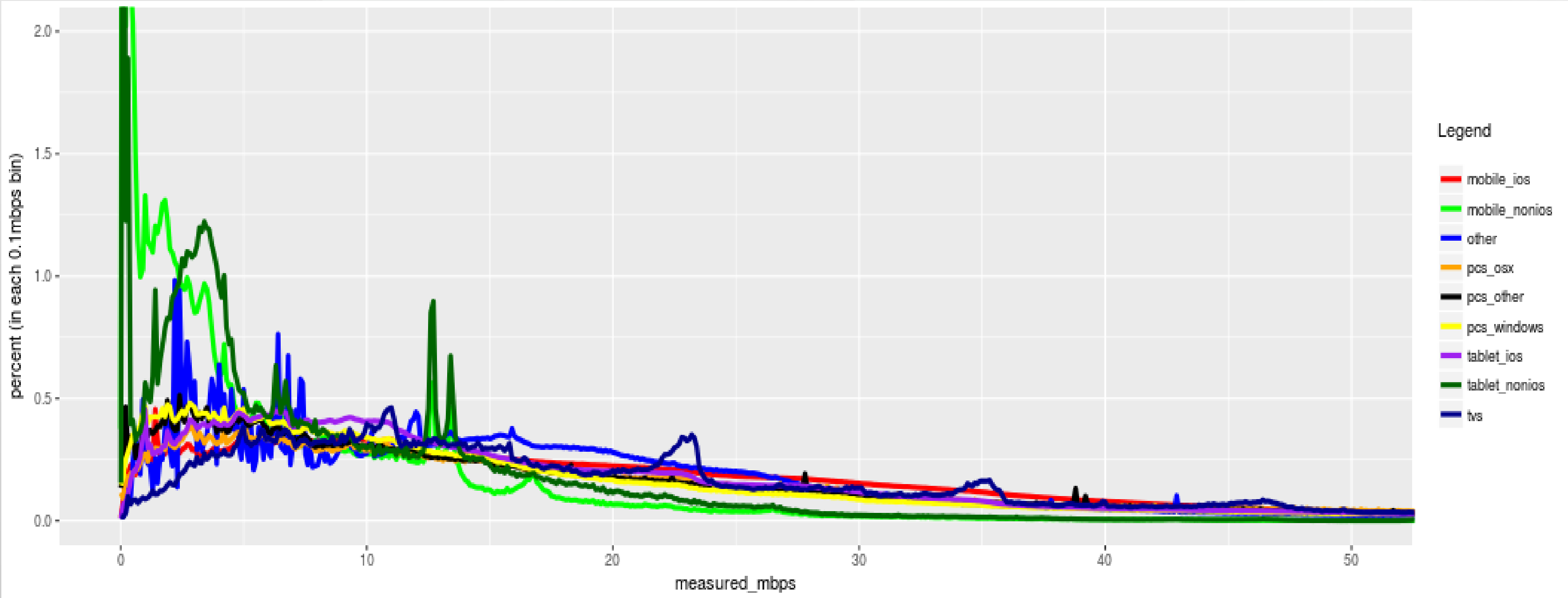
- they provide lists of resolutions and rates that are used for all content, sent to all networks

However, such approach fails to account for differences in characteristics of video content as well as network properties

- differences in video RD performance:
- differences in networks and usage statistics



Source: Netflix, 2015



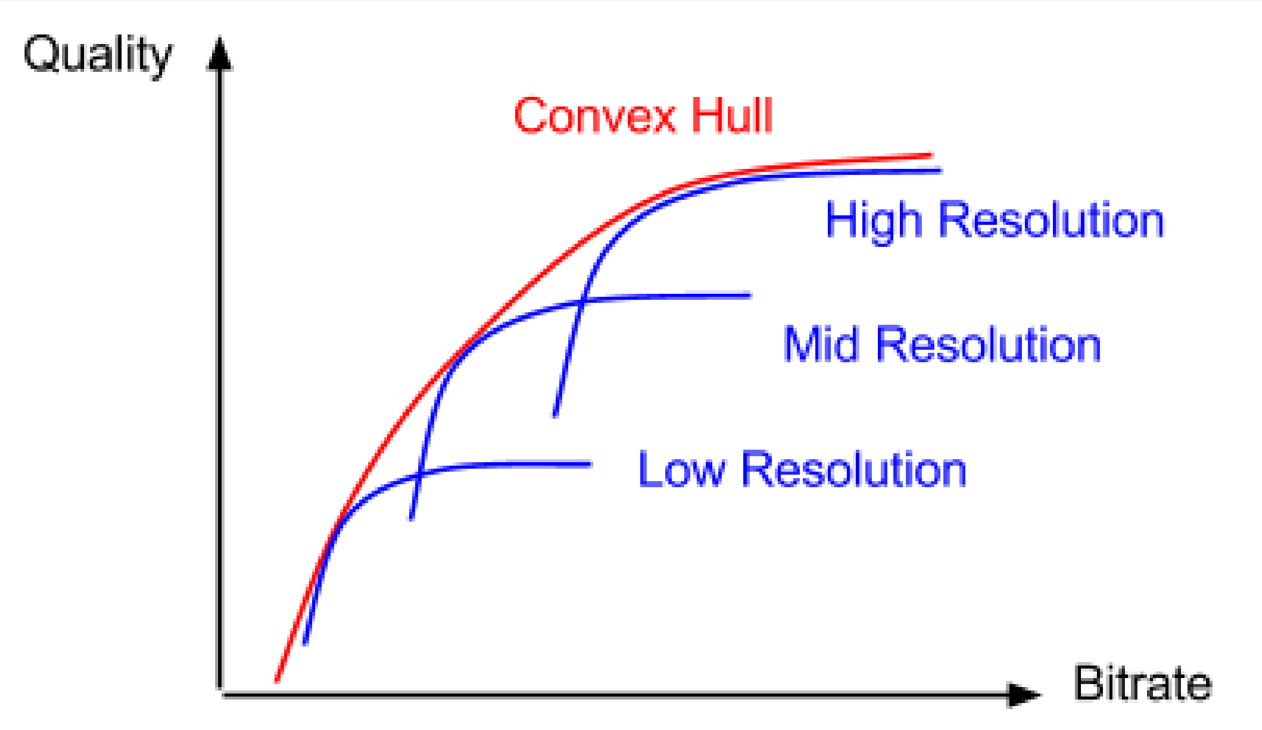
Source: Brightcove VideoCloud analytics, 2017

A better approach is to design encoding ladders **dynamically**, accounting for characteristics of

- content → **content-aware encoding** (aka per-title encoding)
- delivery context/model → **context-aware encoding**

Per-title / Content-aware encoding

As noted by Netflix, when each title is encoded, this produces a composition of quality-rate functions for each resolution



and where the upper boundary of such functions form a convex hull.

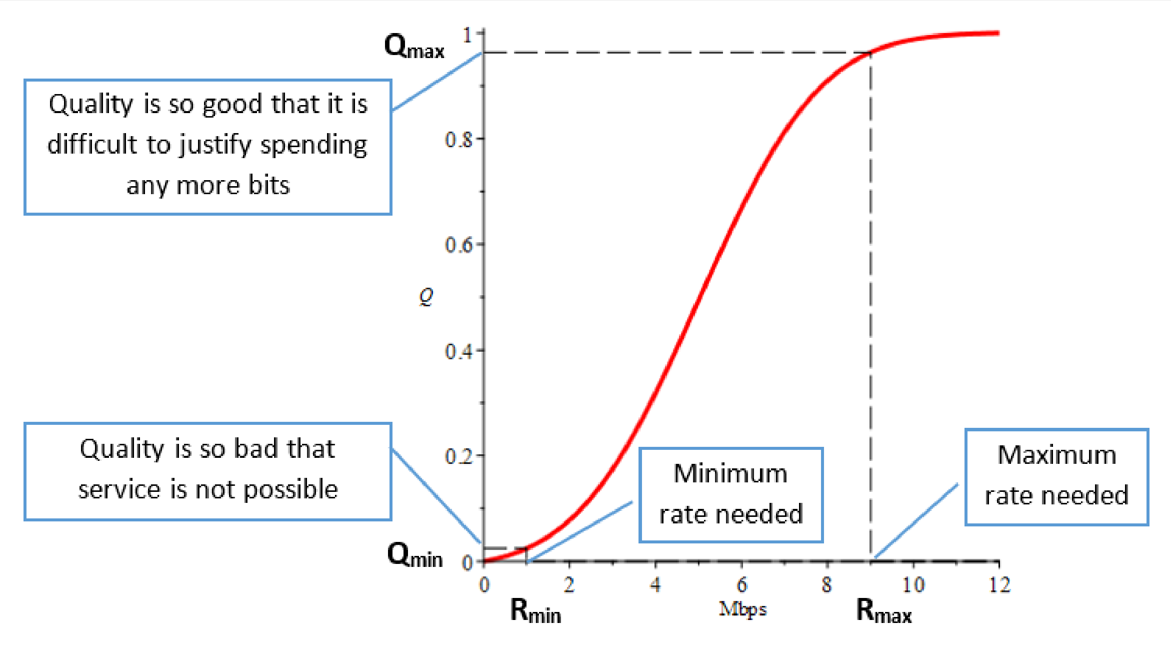
The main idea of per-title encoding is to **pick ladder points such that they belong to the convex hull.**

This provides a method for finding best resolutions for any given target bitrate, but it does not, however, say how such bitrates should be placed, or how many of them are needed.

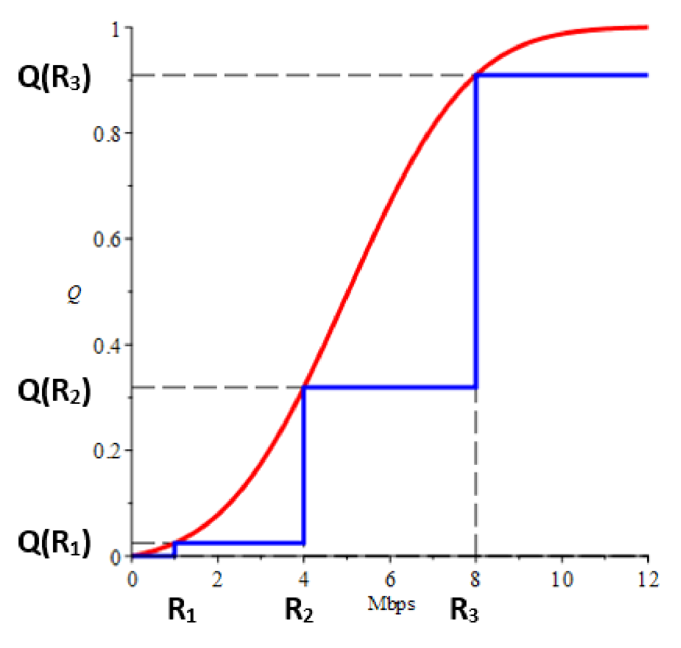
In other words, by itself, “per-title” approach does not result in a fully formed optimization problem!

Context-aware encoding = average quality optimization problem

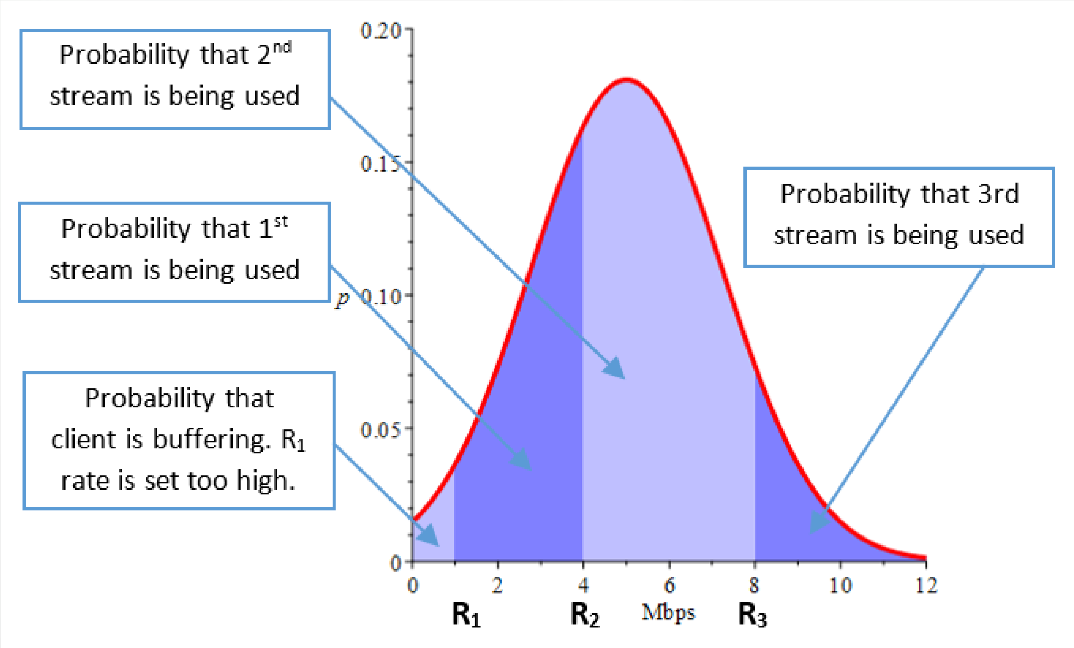
Quality-rate function $Q(R)$:



Quality at each encoding point:



Probabilities of loading each stream:



Given a ladder of rates $R_1, \ldots, R_n$, quality-rate function $Q(R)$, and network PDF $p(R)$, we can define:

– buffering probability: $p(R < R_1) = \int_0^{R_1} p(R)dR$ (probability that playback is not possible, even at lowest rate)

– average quality: $\overline{Q}(R_1, \ldots, R_n, p) = Q(R_1) \int_{R_1}^{R_2} p(R)dR + Q(R_2) \int_{R_2}^{R_3} p(R)dR + \ldots + Q(R_n) \int_{R_n}^{R_{\max}} p(R)dR$

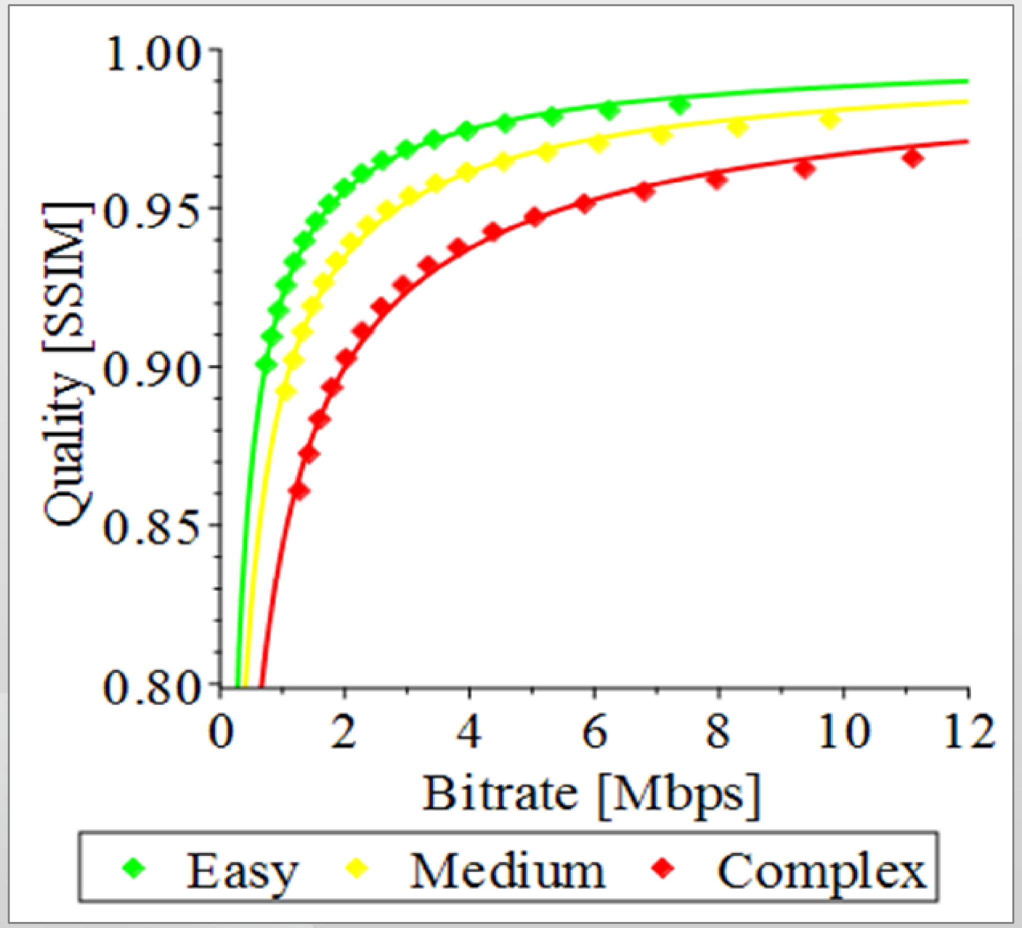
A **quality-optimal profile** is set of rates $R_1^*, \ldots, R_n^*$, such that:

$$\overline{Q}(R_1^*, \ldots, R_n^*, p) = \max_{\substack{R_{\min} < R_1 \leq \cdots \leq R_n < R_{\max} \\ R_1 \leq R_{1,\max}}} \overline{Q}(R_1, \ldots, R_n, p) .$$

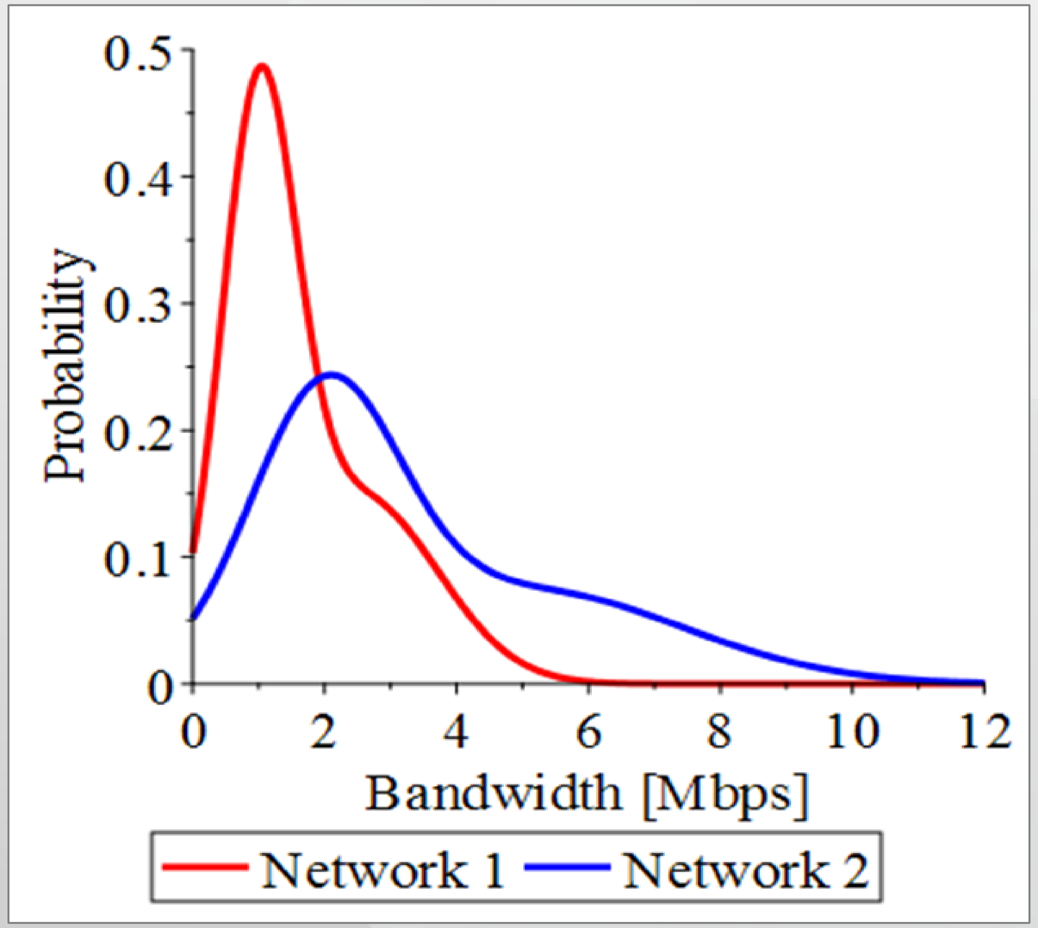
An experiment

Content:

Resolution=720p25
Codec=H264
Quality metric=SSIM
3 sequences:
“Easy”, “Medium”,
“Complex”



Networks:



Based on data from:
J. Karlsson, and
M. Riback. Initial field performance measurements of LTE, *Ericsson review*, 3, 2008.

Quality-rate models:

$$Q(R) = \frac{R^\beta}{\alpha^\beta + R^\beta}$$

Content	α	β
Easy	0.0555	0.8550
Medium	0.0724	0.8016
Complex	0.1015	0.7364

Network models: $p(R) = \alpha\, \mathcal{N}_{\mu_1,\sigma_1}(R) + (1-\alpha)\, \mathcal{N}_{\mu_2,\sigma_2}(R)$

Network	α	μ_1	σ_1	μ_2	σ_2
Network 1	0.584	0.996	0.564	2.554	1.165
Network 2	0.584	1.992	1.129	5.108	2.331

Optimal profiles for given source and network models

Optimal profiles for Network 1:

Content	N	Profile bitrates [kbps]	Q_n	$\bar{Q}$	ξ [%]
Easy	2	138, 803	0.909	0.867	6.58
	3	100, 512, 1209	0.931	0.888	4.35
	4	100, 411, 866, 1645	0.946	0.897	3.34
	5	100, 349, 694, 1155, 2087	0.955	0.902	2.76
Medium	2	175, 854	0.881	0.830	7.98
	3	100, 518, 1219	0.906	0.854	5.31
	4	100, 416, 876, 1663	0.924	0.866	4.00
	5	100, 354, 701, 1165, 2104	0.936	0.873	3.25
Complex	2	234, 931	0.825	0.769	10.2
	3	145, 590, 1304	0.867	0.797	6.96
	4	102, 431, 898, 1704	0.888	0.812	5.22
	5	100, 363, 716, 1183, 2134	0.904	0.821	4.16

Optimal profiles for Network 2:

Content	N	Profile bitrates [kbps]	Q_n	$\bar{Q}$	ξ [%]
Easy	2	232, 1457	0.940	0.906	5.14
	3	116, 811, 2124	0.955	0.924	3.27
	4	100, 589, 1421, 2803	0.964	0.932	2.40
	5	100, 486, 1107, 1974, 3577	0.971	0.937	1.92
Medium	2	293, 1549	0.920	0.878	6.23
	3	158, 893, 2216	0.939	0.899	4.04
	4	100, 601, 1438, 2828	0.949	0.909	2.97
	5	100, 495, 1123, 1995, 3615	0.958	0.915	2.35
Complex	2	391, 1685	0.887	0.833	7.98
	3	232, 1018, 2358	0.910	0.857	5.29
	4	156, 712, 1569, 3001	0.924	0.869	3.94
	5	114, 537, 1179, 2060, 3727	0.935	0.877	3.11

Q_n = quality at top rendition [SSIM]

$\bar{Q}$ = average quality [SSIM]

ξ = gap to quality achievable with infinite number of renditions [%]

$$\bar{Q} = Q(R_1)\int_{R_1}^{R_2} p(R)dR + Q(R_2)\int_{R_2}^{R_3} p(R)dR + \cdots + Q(R_n)\int_{R_n}^{R_{\max}} p(R)dR\,,$$

$$Q_n = Q(R_n), \qquad Q^* = \int_0^{\infty} Q(R)\,p(R)dR, \qquad \xi = \frac{Q^* - \bar{Q}}{Q^*}\cdot 100\ [\%]$$

Key observation:

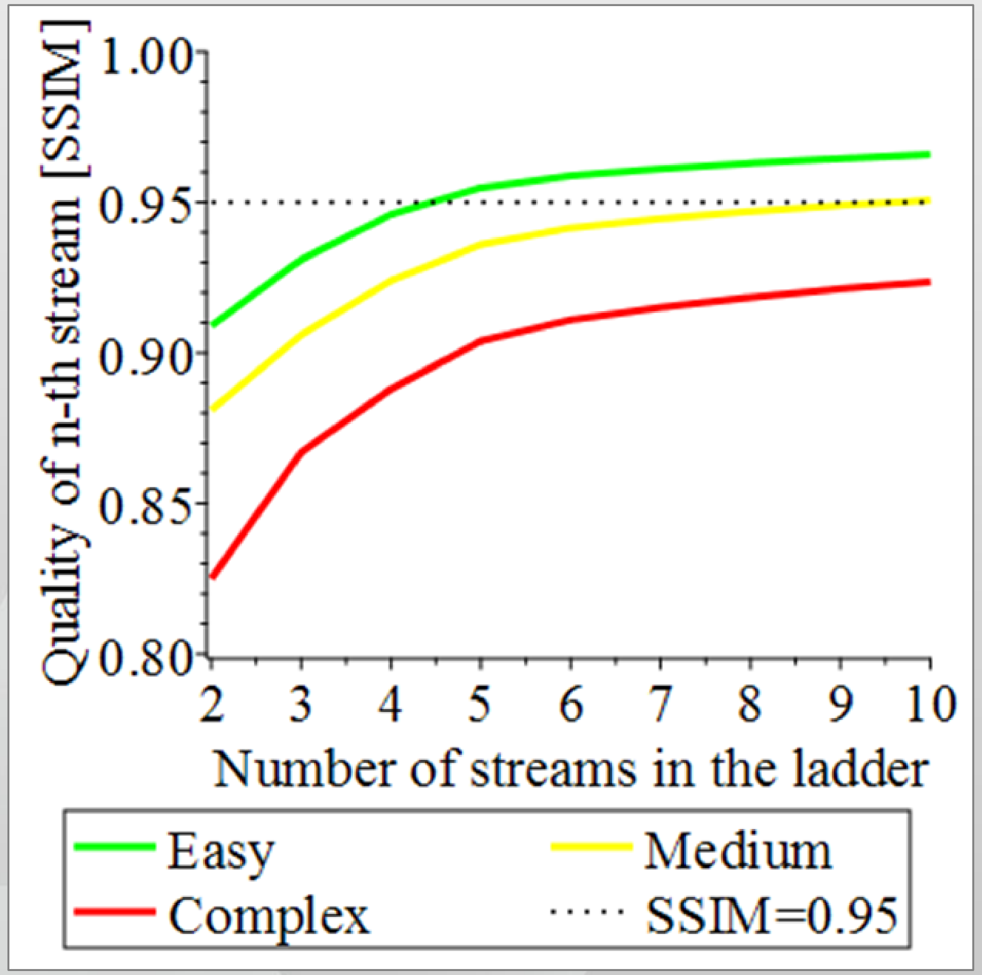
➔ **optimal profiles designed for different sources and networks are different!**

brightcove

Sufficient number of encoding points

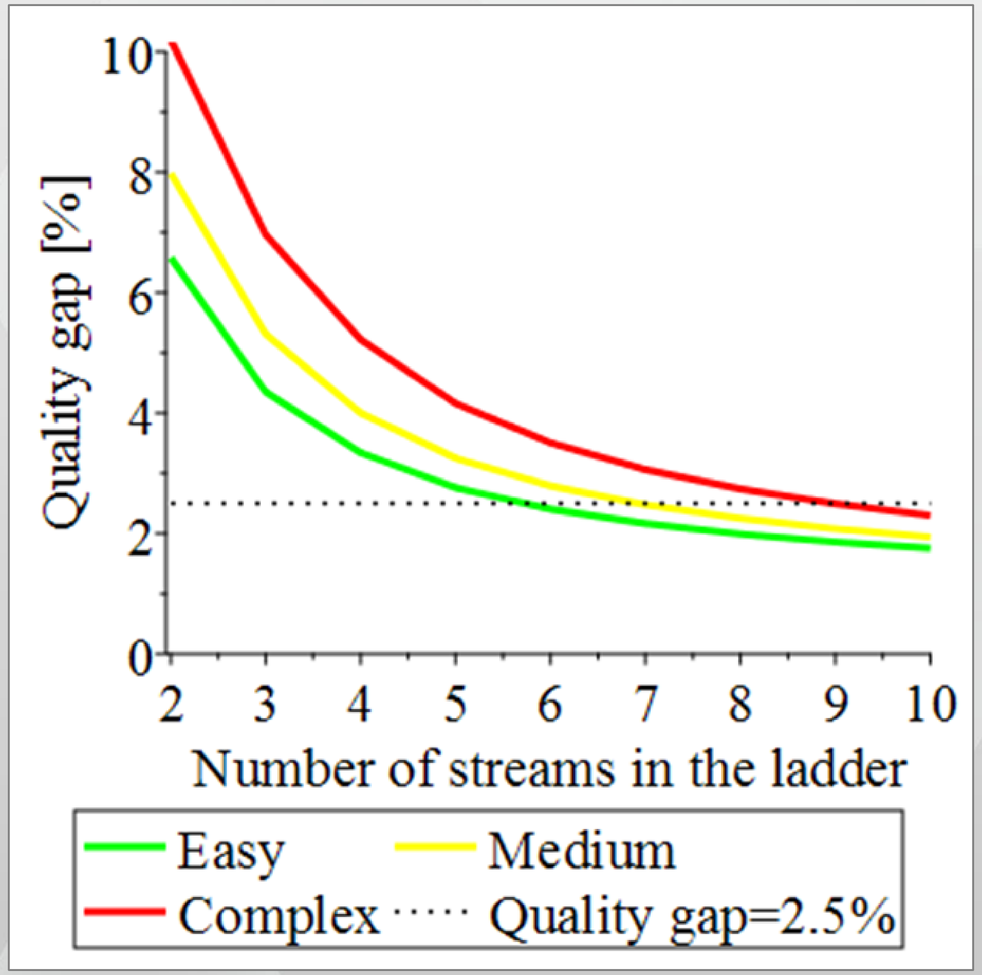
There are 2 criteria that can be utilized:

(1) Limit for quality at top rendition:



This shows that "easy" content can be encoded with much fewer renditions!

(2) Limit for quality gap:



This provides effective bound on the number of renditions for "complex" content as well.

brightcove

Quick summary

As shown earlier, given quality-rate function $Q(R)$, information about network $p(R)$, and client model, we can define the problem of design of encoding ladder as problem of **maximizing average quality delivered to the clients**:

$$\overline{Q}(R_1^*, \dots, R_n^*, p) = \max_{\substack{R_{\min} < R_1 \le \cdots \le R_n < R_{\max} \\ R_1 \le R_{1,\max}}} \overline{Q}(R_1, \dots, R_n, p)$$

This problem clearly belongs to a class of **non-linear constrained optimization problems.**

In cases when average quality function $\overline{Q}(R_1, \dots, R_n, p)$ is differentiable w.r.t. $R_1, \dots, R_n$ this problem is well known and can be solved by using existing numerical optimization techniques, such as **sequential quadratic programming**.

Further more, by using additional limits for quality of top rendition $Q(R_n)$, as well as quality gap $\xi(R_1, \dots, R_n, p)$ we can also bound the number of encoding points such that overall performance stays close to optimal.

In other words, the problem optimal design of encoding profiles is now fully defined.

Why use network statistics?

Q: Isn't it the case that the whole purpose of ABR streaming is to enable operation regardless of network characteristics?

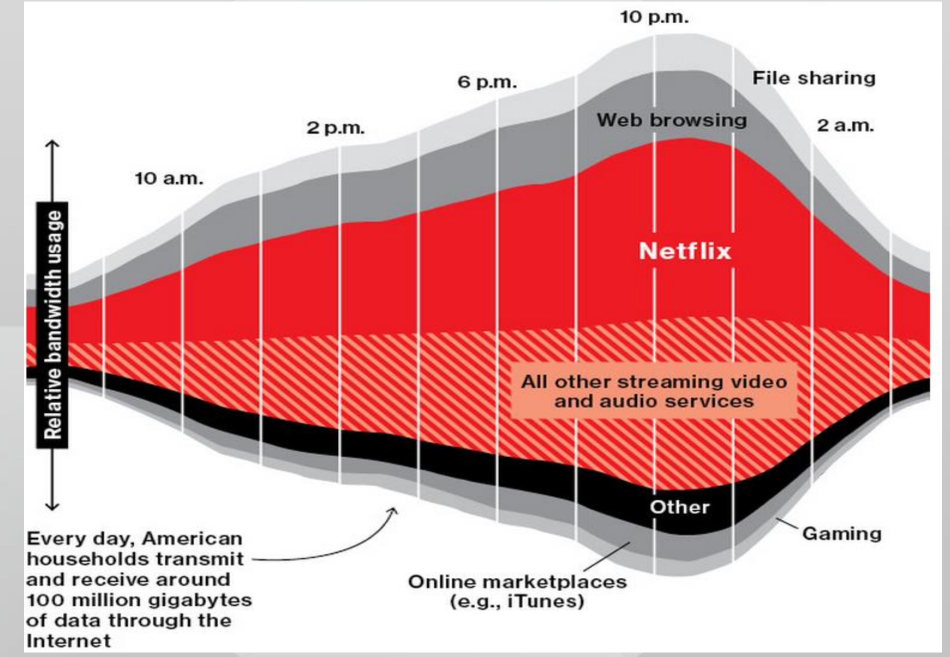
A: Yes, and No.

Yes, the basic objective of ABR was to enable continuous playback if bandwidth is unknown or changing

No: each operator **knows a lot** about its networks and users, and not using such statistics for improving quality of service is a crime! Especially for live events or services with known geographic distribution of users.

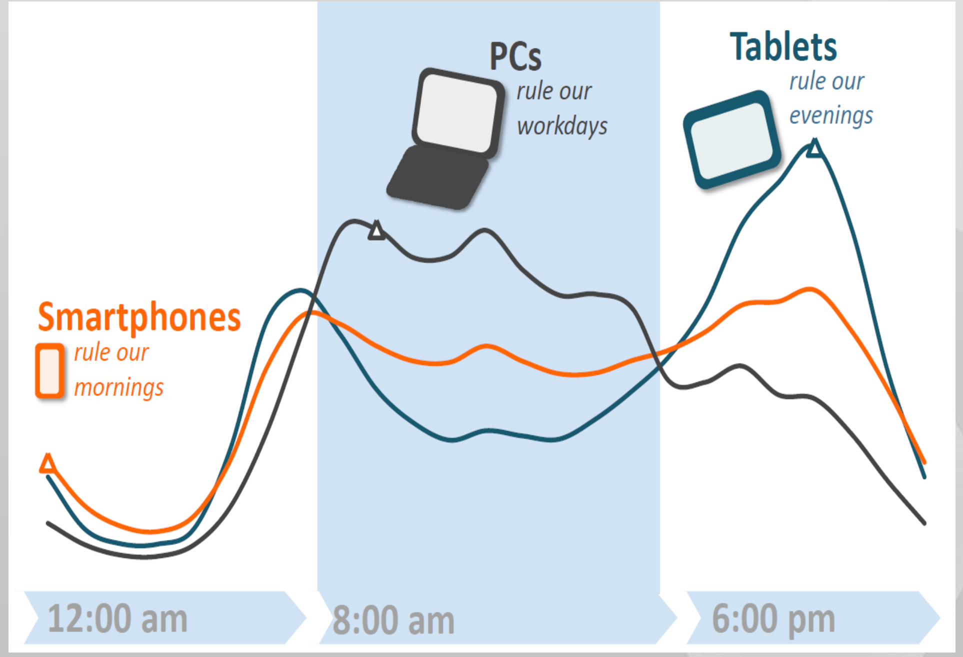
Examples of well-known networks- and usage- related phenomena:

Changes of network traffic:



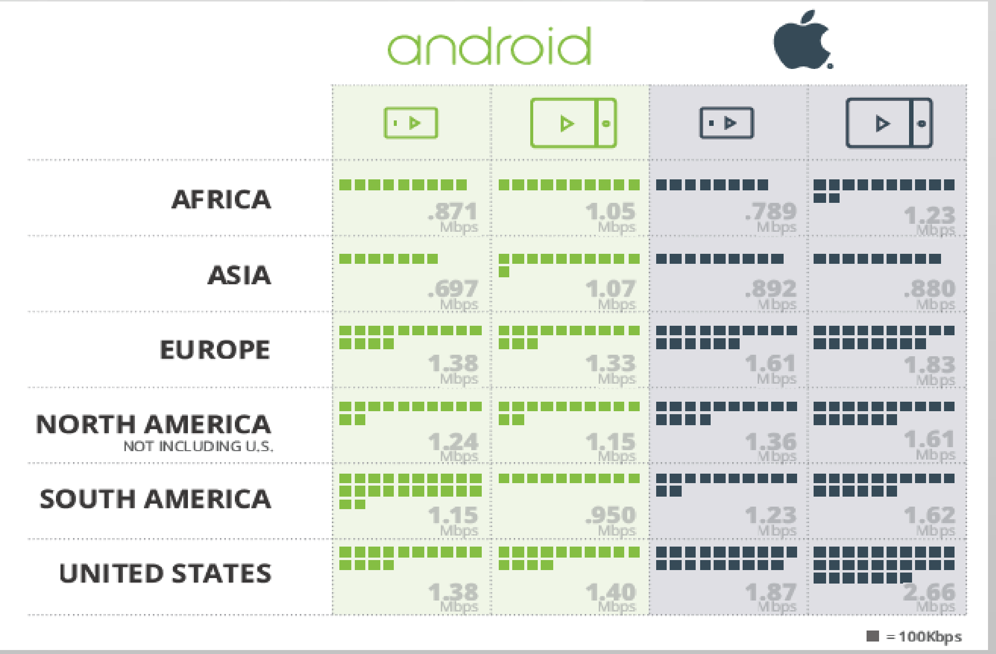
Source: *Bloomberg BusinessWeek, May 5th 2013*

Changes of usage of devices/screens:



Source: *comScore, February 2013*

Network bandwidth per region:



Source: *Conviva VXR, 2015*

brightcove

Generalizations and extensions

Given:

$R_1, \dots, R_n$ – list of ladder bitrates,

$Q(R)$ – quality-rate function,

$p(R)$ – network PDF, and

$R_{\text{selected}}(B) = f(B, R_1, \dots, R_n, p)$ – client model

We can compute probabilities of loading of each stream, and subsequently define and analyze average performance parameters of the streaming system. Average behavior of streaming system becomes fully characterized.

Moreover, for **any client**, we may expect that

$R_{\text{selected}}(B) \to f(B, R_1, \dots, R_n, p)$ (pr.)

so most results will hold.

Generalizations to configurations with multiple networks, devices, codecs, resolutions, etc. are also easily derivable.

Parameter	**Expression**
Average bandwidth used for streaming	$\bar{R}(p, R_1, \dots, R_n) = \sum_{i=i}^{n} p_i R_i$
Average network bandwidth	$\bar{B}(p) = \int_0^\infty R\, p(R)\, dR$
Bandwidth utilization	$\eta(p, R_1, \dots, R_n) = \frac{\bar{R}(p, R_1, \dots, R_n)}{\bar{B}(p)}$
Buffering probability	$p_0(p, R_1) = \int_0^{R_1} p(R) dR$
Average quality	$\bar{Q}(p, R_1, \dots, R_n) = \sum_{i=1}^{n} p_i Q(R_i)$
Average quality limit	$Q^*(p) = \int_0^\infty Q(R)\, p(R)\, dR$
Quality gap	$\xi(p, R_1, \dots, R_n) = \frac{Q^*(p) - \bar{Q}(p, R_1, \dots, R_n)}{Q^*(p)}$

Thank you!

Contact:
Yuriy Reznik
Brightcove, Inc
yreznik@brightcove.com