BRIGHTCOVE®

ON MULTIPLE MEDIA REPRESENTATIONS AND CDN PERFORMANCE

Yuriy Reznik, Thiago Teixeira, Robert Peck
Brightcove, Inc.

ACM Mile-High Video
March 1, 2022

© Brightcove Inc. All Rights Reserved.

BRIGHTCOVE

Outline

Introduction

- Evolution of streaming. First adaptive multi-rate systems
- Modern-era HTTP-based streaming systems
- The disconnect between multi-rate streaming and CDN model

Modeling CDN cache performance

- Content popularity model
- Cache miss probability in case of single content representation
- Cache miss probability in case of multiple content representations
- Relative effect of multiple representations

Experimental validation

- Experiment setup
- The results

Extensions and concluding remarks

© Brightcove Inc. All Rights Reserved.

BRIGHTCOVE®

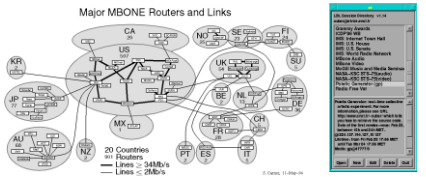
Introduction

© Brightcove Inc. All Rights Reserved.

Early Days of Streaming

1993: MBONE

- Virtual multicast network connecting several universities & ISPs
- RTP-based video conferencing tool (vic) is used to send videos
- 1994 Rolling Stones concert – first major event streamed online



1995: RealAudio, 1997: RealVideo

- First commercially successful mass-scale streaming system
- Proprietary protocols, codecs: PNA, RealAudio, RealVideo
- Worked over UDP, TCP, and HTTP (“cloaking” mode)
- First major broadcast: 1995 Seattle Mariners vs New York Yankees

1995+: VDOnet, Vivo, NetShow, VXtream, ...

- Many vendors have tried to compete in streaming space initially
- Vivo & Xing got acquired by Real, VXtreme by Microsoft
- By 1998, 3 main vendors remained: Real, Microsoft and Apple

1998: RealSystem G2

- First ABR streaming system

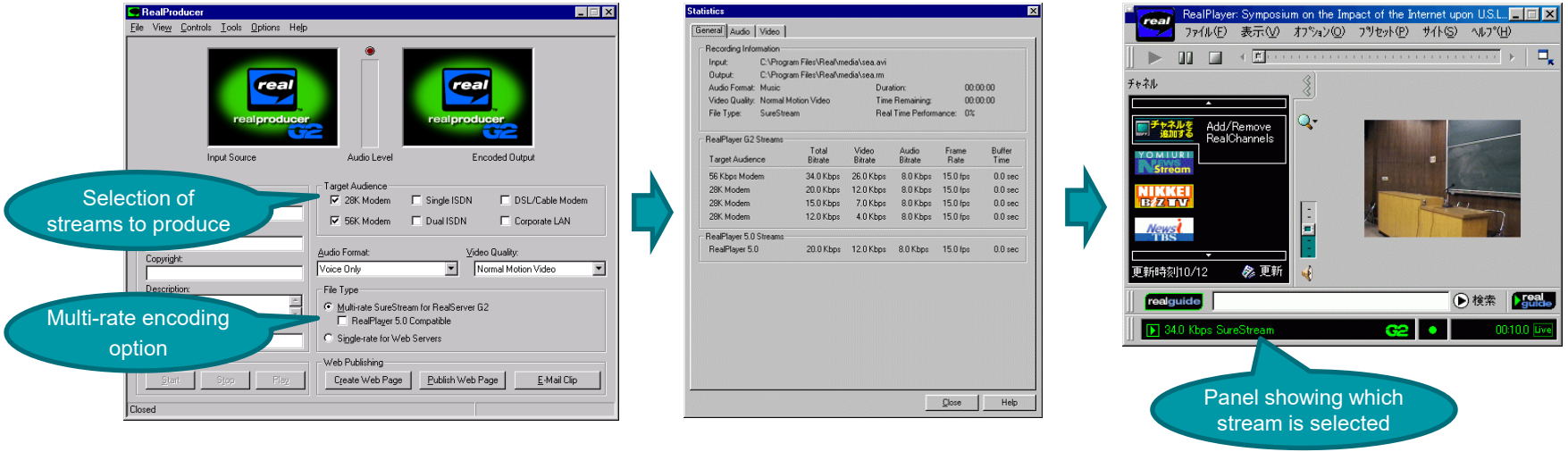
First ABR Streaming System

1998: RealSystem G2: “SureStream”

- First commercially successful ABR streaming system
- Encoder:

Encoded streams

Player

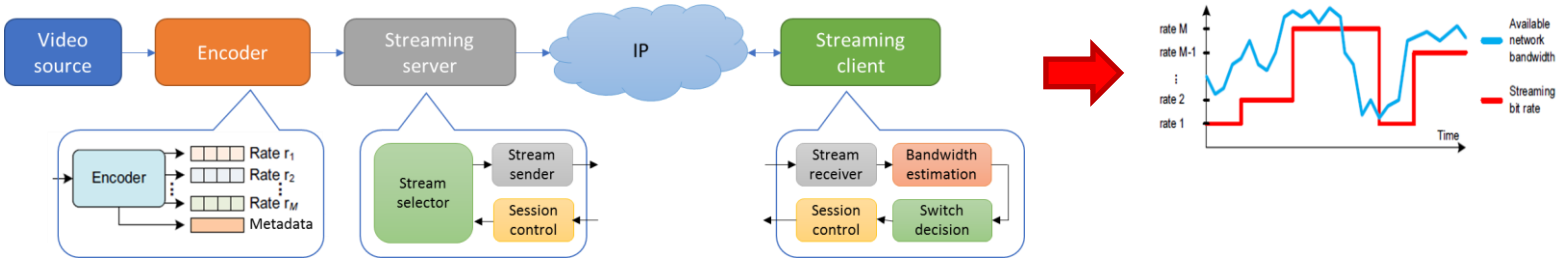


Related publications & patents

- B. Girod, et al, “Scalable codec architectures for Internet video-on-demand,” ACSSC, pp. 357 – 361, 1997.
- G. Conklin, et al, “Video Coding for Streaming Media Delivery on the Internet," *TCSVT*, 11 (3), pp. 20-34, 2001.
- US Patents: 6314466, 6480541, 7075986, 7885340

How First ABR System Worked?

RTSP/UDP-based streaming architecture:



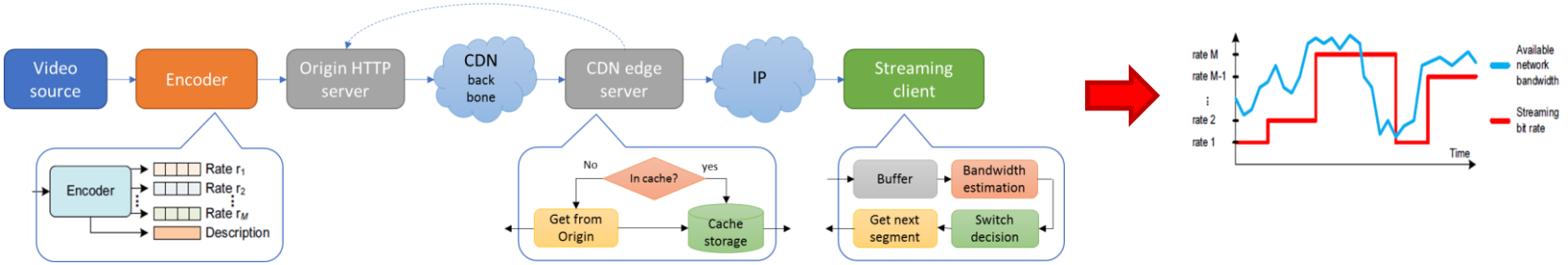
- Public internet is used for delivery
- RTSP protocol was used for session control, and UDP (plus RTP or proprietary transport) were used for sending the data
- Stream adaptation was done by server, but with most clients – it was client-driven: client was sending requests to switch
- Server was also responsible for retransmissions, injecting extra FEC packets, etc.
- Everything was sent in “packets”

Important design elements:

- **Only one stream** was sent of over IP for delivery to each client!
- Multiple renditions were stored only on the (origin) streaming server, and transmissions of such “stacks of streams” to other servers was not even envisioned.
- This was all before CDNs and relay networks for streaming!

HTTP-based ABR Streaming

Modern-era HLS/DASH architecture:



Key differences from RTSP/UDP streaming:

- instead of streaming server, a regular HTTP server is used as origin
- stream switching is trivialized to HTTP GET operations originating from streaming client
- the scaling and delivery is delegated to CDN, which caches content on the edge servers, reducing the load on the origin...

Important new factors:

- This works well when the “content” is popular and it becomes stored in the edge cache
- If content is not popular, and not stored at the edge cache – it becomes pulled from the origin server (in which case CDN only adds latency and increases cost of delivery)
- In other words – optimistically CDN helps, but in the worst case – it does not!

Disconnect between ABR and CDN models

Key issues:

- ABR systems fundamentally need **several encoded versions of the content:**
  - Multiple streams are needed to achieve better network adaptation and minimize the visibility of stream switches.
  - Multiple streams are also needed to support different delivery formats (HLS, DASH, MSS, etc.) and DRM systems.
  - Support for multiple video codecs (H.264, HEVC, AV1, and VVC) also results in a creation of multiple streams

- However, once multiple streams are created, and different client start pulling different versions of then – such streams start **"competing" for the CDN edge cache disk space.** This results in mode CDN cache misses, and higher load on origin server. This also increases delivery costs and makes whole system less reliable, less scalable, etc.

- In other words, while **ABR streaming concept promotes the creation of "more" streams, what CDNs need to be the most effective is "less"!**

Objectives of this talk:

- Offer few mathematical models quantifying the impact of multiple streams/representations on CDN performance
- Offer recommendations for the design of ABR systems to make them more efficient from the CDN performance point of view

BRIGHTCOVE®

Modeling CDN performance

© Brightcove Inc. All Rights Reserved.

Content popularity model

Let us assume that:

- We have a set of items (e.g. videos or segments)

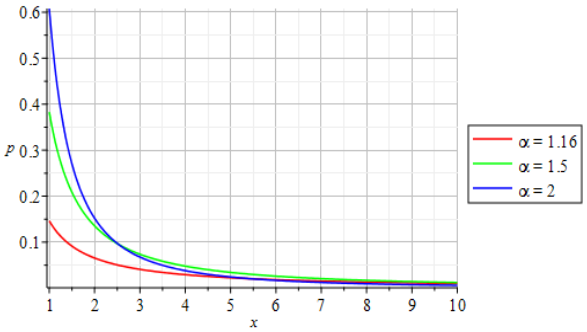
$$S = \{s_x, x \geq 1\},$$

- These items are sorted in the order of decreasing probabilities of their use
- And that their probabilities in such order can be described by some model
- Specifically, we will assume that they can be modeled by Zeta distribution:

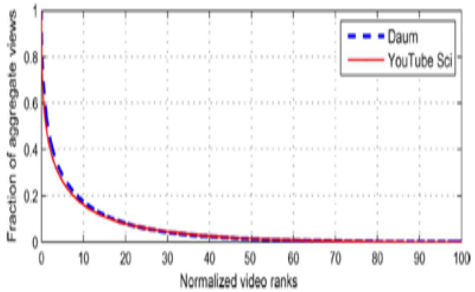
$$p(x) = \frac{x^{-\alpha}}{\zeta(\alpha)}$$

- where where α is a shape parameter, and $\zeta(.)$ is a Riemann Zeta function
- This is a classic discrete distribution model, with many known examples of its used in similar contexts. E.g. it is known to provide a good approximation of popularity of videos on YouTube.

Shape of popularity distribution model for different values of parameter α



Empirically measured popularity of videos on YouTube



M. Cha, H. Kwak, P. Rodriguez, Y-Y. Ahn, S. Moon, Sue. (2009). Analyzing the Video Popularity Characteristics of Large-Scale User Generated Content Systems. IEEE/ACM Trans. Netw.. 17. 1357-1370.

Idealized cache model

Let us further assume that:

- We have a cache with **capacity of C** items
- And this cache works such that it knows exactly probabilities of all items that may be placed in it, and then it only **stores C items which have highest probabilities of occurrence**

Then:

- By considering that input items follow Zeta distribution:

$$p(x) = \frac{x^{-\alpha}}{\zeta(\alpha)}$$

- The probability that any randomly selected item x falls outside of cache of size C will be

$$p_{miss}(C, \alpha) = \mathbf{1} - \sum_{x=1}^{C} p(x) = \mathbf{1} - \frac{H_{C,\alpha}}{\zeta(\alpha)},$$

- where $H_{C,\alpha} = \sum_{x=1}^{C} x^{-\alpha}$ is a generalized Harmonic number

- When cache size C is large, this asymptotically turns into:

$$p_{miss}(C, \alpha) \sim \frac{C^{1-\alpha}}{(\alpha - 1)\zeta(\alpha)}\left(1 + O\left(\frac{1}{C}\right)\right)$$

- Nice and simple formula!

Cache model with 2 sets of items

Let us next assume that:

- We have 2 sets of content items: $S_1 = \{s_{1,x}, x \geq 1\}$, and $S_2 = \{s_{2.x}, x \geq 1\}$
- Relative usage probabilities of these 2 sets will be denoted as $\pi = \{\pi_1, \pi_2\}$
- Then, the full probabilities of items in each set become:

$$p(s_{1,x}) = \pi_1 \cdot p(s_x) \quad \text{and} \quad p(s_{2,x}) = \pi_2 \cdot p(s_x)$$

Structure at the top of the cache (case when $\boldsymbol{\pi_1 > \pi_2}$):

Item	Probability	Comments
$s_{1,1}$	$\pi_1 p(1)$	First go items packaged using more widely supported format
...	...	...
$s_{1,x}$	$\pi_1 p(x)$	$x = \left\lfloor \left(\frac{\pi_1}{\pi_2}\right)^{\frac{1}{\alpha}} \right\rfloor$, solution of $\pi_1 p(x) = \pi_2 p(1)$
$s_{2,1}$	$\pi_2 p(1)$	
$s_{1,x+1}$	$\pi_1 p(x+1)$	Then follow items from more widely supported content
...	...	...
s_{1,x_2}	$\pi_1 p(x_2)$	$x_2 = \left\lfloor 2\left(\frac{\pi_1}{\pi_2}\right)^{\frac{1}{\alpha}} \right\rfloor$, solution of $\pi_1 p(x_2) = \pi_2 p(2)$
$s_{2,2}$	$\pi_2 p(2)$	Next comes the second item packaged in less supported format
s_{1,x_2+1}	$\pi_1 p(x_2+1)$	Then again follow items from more widely supported format
...	...	...

NB: Items from less frequently used set **become injected in this stack with step size of $x \sim (\pi_1/\pi_2)^{1/\alpha}$** !

Cache miss probability with 2 sets

- Asymptotically with large cache size C and 2 versions of the content, the cache miss probability becomes:

$$p_{miss,2}(C,\alpha,\pi) \sim \left(\pi_1^{\frac{1}{\alpha}} + \pi_2^{\frac{1}{\alpha}}\right)^{\alpha} \frac{C^{1-\alpha}}{(\alpha-1)\zeta(\alpha)}\left(1 + O\left(\frac{1}{C}\right)\right)$$

- This looks similar to cache miss probability in case of singe set/representation:

$$p_{miss}(C,\alpha) \sim \frac{C^{1-\alpha}}{(\alpha-1)\zeta(\alpha)}\left(1 + O\left(\frac{1}{C}\right)\right)$$

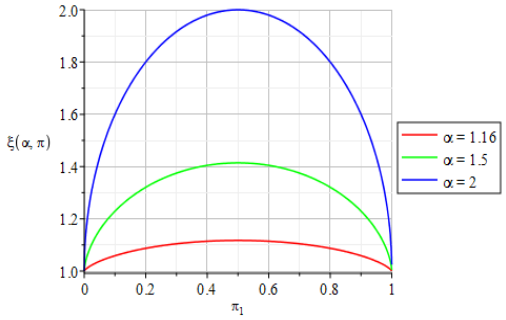
- Further, if we next look at the ratio:

$$\xi(\alpha,\pi) = \frac{p_{miss,2}(C,\alpha,\pi)}{p_{miss}(C,\alpha)} \sim \left(\pi_1^{\frac{1}{\alpha}} + \pi_2^{\frac{1}{\alpha}}\right)^{\alpha}$$

- We discover that it becomes asymptotically independent on C!

- In other words, considering any CDN with reasonably large cache, we can predict that the use of 2 versions will increase its cache miss probability by $\left(\pi_1^{1/\alpha} + \pi_2^{1/\alpha}\right)^{\alpha}$

Relative increase in cache miss probability in case of using 2 formats.



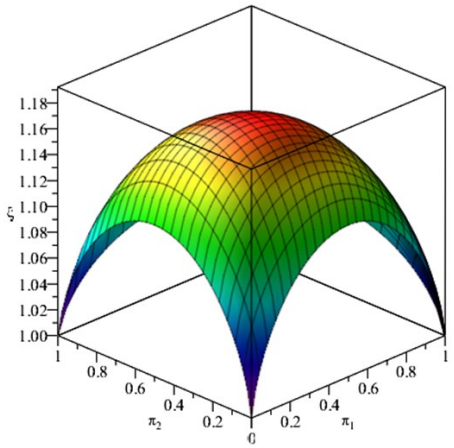
Cache miss probability with k sets

- More generally, it can be shown, that asymptotically (with large CDN cache size) the use of k versions will increase its cache miss probability by a factor of

$$\xi(\alpha,\pi)=\frac{p_{miss,k}(C,\alpha,\pi)}{p_{miss}(C,\alpha)}\sim\left(\sum_{i=1}^{k}\pi_i^{\frac{1}{\alpha}}\right)^{\alpha}=\|\pi\|_{\frac{1}{\alpha}}$$

- Where α is a parameter of content popularity model, and $\pi=\{\pi_1,\dots,\pi_k\}$ are the usage probabilities of each format

Relative increase in cache miss probability in case of using 3 formats.



Observations

- the worst impact happens when all formats are equally probable:

$$\pi_1=\cdots=\pi_k$$

- the higher is the asymmetry in usage of different formats (or renditions), the better it is from CDN efficiency standpoint:

$$\pi_i\to 1\Rightarrow\ \xi(\alpha,\pi)\to 1$$

Recipe for success:

- To improve CDN performance with multiple representations/formats - pick one “preferred” representation, and direct as many possible clients/devices use it!

BRIGHTCOVE®

Experiments

© Brightcove Inc. All Rights Reserved.

Experiment Setup

Comparing HLS vs 2-format HLS+DASH deployments

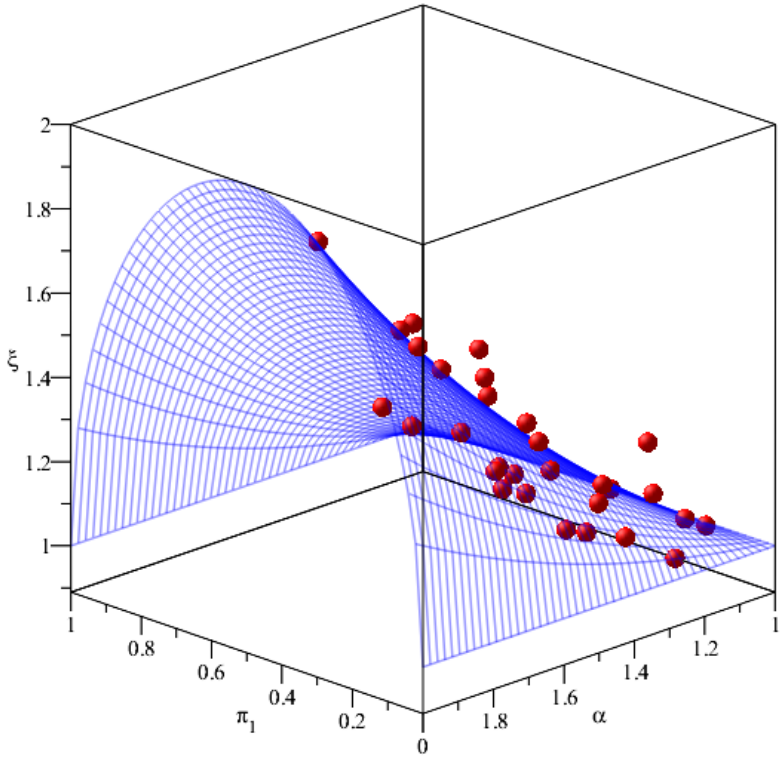
- Using Brightcove VideoCloud system
- Selected 1000 accounts with HLS-only and mixed HLS and DASH deployments
- Out or these 1000 accounts selected 30 account pairs, where
  - One of the accounts is HLS-only, and the other is HLS + DASH
  - The both have similar overall volume of requests, as well as shape of popularity distribution (model parameter α)
- For each pair of systems:
  - Computed CDN-reported cache-miss probabilities p_{miss}, $p_{miss,2}$
  - Computed ratios $\xi = \frac{p_{miss,2}}{p_{miss}}$
  - Also computed ratios as predicted by our model $\xi(\alpha, \pi)$

Overall characteristics of the experiment:

Characteristic	Value
The total number of records in CDN logs used for analysis	40M
The number of pairs of HLS and HLS+DASH systems selected for comparative analysis	30
RMS of fit of α parameters of popularity distributions	0.051
RMS of fit of ξ ratios as predicted by model vs CDN-reported data	0.098

The Results

Plots of model-predicted vs measured changes in CDN performance:



- Blue = model predicted
- Red = measured

- Overall RMS = 0.098

- π - fraction of content pulled in DASH
- α – popularity distribution parameter

BRIGHTCOVE®

Extensions and concluding remarks

© Brightcove Inc. All Rights Reserved.

BRIGHTCOVE®

Conclusions

The results

- Simple models predicting the impact of multiple formats on CDN cache performance
- Shown that the predictions correlate well with data observed in the field

The proposed models can be easily extended

- To cover renditions using different bitrates
- Mixed codec renditions
- SHD and HDR formats, etc.

© Brightcove Inc. All Rights Reserved.

BRIGHTCOVE®

THANK YOU

© Brightcove Inc. All Rights Reserved.