Making Systems Faster: Distributed Video Processing

COS 418/518: (Advanced) Distributed Systems
Lecture 19
Mike Freedman & Wyatt Lloyd

[Grey slides from Qi Huang's SOSP 2017 Talk]

Distributed Video Processing Outline

- Motivation for video processing
  - (How streaming video works)
- Legacy design
- SVE design
- Why SVE is faster than legacy

SVE: Distributed Video Processing at Facebook Scale

Qi Huang
Petchean Ang, Peter Knowles, Tomasz Nykiel, Iaroslav Tverdokhlib, Amit Yajurvedi, Paul Dapolito IV, Xifan Yan, Maxim Bykov, Chuen Liang, Mohit Talwar, Abhishek Mathur, Sachin Kulkarni, Matthew Burke, Wyatt Lloyd

Facebook, University of Southern California, Cornell, Princeton

Video is growing across Facebook

- FB: **500M** users watch **100M hours** video daily (Mar. 16)
- Instagram: **250M** daily active users for stories (Jun. 17)
- All: **many tens of millions** of daily uploads, **3X** NYE spike
Processing is diverse and demanding

Pt. 1
Legacy System
Scaling Challenges

Pt. 2
SVE
Impact of Design

Legacy: upload video file to web server

Legacy: preserve original for reliability

Legacy: process after upload completes
She is having so much fun with #MSQRD

Legacy: encode w/ varying bitrates

06

Legacy: store encodings before sharing

07

Sharing with adaptive streaming

08

Focus: pre-sharing pipeline

09

All steps from when a user starts an upload until a video is ready to be shared
How Long Does This Take? (Latency)

Client ➔ Web Server ➔ Original Storage ➔ Processing ➔ Final Storage

- 16 MB Video ≈ 16 secs
  - 1 Mbps link

- 1 MB Video ≈ 1 sec
  - 8 Mbps link

SVE paper stats:

- Video Size
  - ≤1MB: 10% of uploads over 10 seconds
  - 1-10MB: 50% of uploads over 10 seconds
  - 300MB-1GB: 50% of uploads over 9 minutes

How Long Does This Take? (Latency)

Web Server ➔ Original Storage

(pipelined with uploading)

SVE paper stats:

- median: 200 ms
- 90%: 650 ms
- 99%: 900 ms
How Long Does This Take? (Latency)

SVE paper stats:
10% of all video take ≥ 1.3 s
Proportional to video size:
Most videos over 100 MB take over 6 seconds

Serial pipeline leads to slow processing

Let’s Make This Faster!
Talk to classmates about how!
**Speedy: harness parallelism**

Users can share videos quickly

- Overlap fault tolerance and processing
- Overlap upload and processing
- Parallel processing

**Architectural changes for parallelism**

1. Client → Web Server → Original Storage → Processing → Final Storage
2. Client → Web Server → Preprocessor → Worker → Final Storage
3. Client → Web Server → Cache → Worker → Final Storage
Let’s Make This Faster!

We decrease overall latency by overlapping uploading and processing (§4.2). The time required to process a video (D–E) is a significant part of the pre-sharing latency. Figure 4b shows CDFs of standard definition (SD) encoding time for different size classes of videos under MES. Unsurprisingly, there is a strong correlation between video size and encoding time. For the size classes smaller than 10 MB, most videos can be encoded in fewer than 10 seconds. Yet, for even the smallest 3–10 MB size class, more than 2% of videos take 10 or more seconds to encode. For large videos the encoding time is even more significant, with 23% of videos larger than 1 MB approximately 10% of uploads take more than 10 seconds. For the large size classes of 30–100 MB, 100–300 MB, and 500–1000 MB, more than half of the uploads take more than 10 minutes. This demonstrates that processing time is a significant part of pre-sharing latency.

We decrease the latency for uploads through client-side re-encoding of the video to a smaller size when three conditions are met: the large and diverse set of clients devices that upload videos; the large and diverse set of devices that can support client-side processing; and the availability of device resources (e.g., battery) since they will provide little benefit. Requiring all three conditions ensures we only do client-side re-encoding when it meaningfully decreases pre-sharing latency. Our insight is to opportunistically use client-side processing to enable faster sharing when it is possible and helpful, but not when it would use up the client-side resources. Fortunately, segmenting a video along GOP boundaries can be shared over News Feed or sent over Messenger. This makes processing of the video parallelizable. Each segment can be processed separately from, and in parallel with, each other segment. The challenges here are in selecting a segment size, enabling per-segment encoding, and ensuring the resulting video is still well formed.
Results: 2.3x ~ 9.3x speedup

Results: 2.3x ~ 9.3x speedup

Results: 2.3x ~ 9.3x speedup

Summary

- Motivation for video processing
  - (How streaming video works)

- Legacy design – Serial processing was slow

- SVE design – Three sources of parallelism make SVE faster
  - Overlap upload and processing
  - Overlap fault tolerance and processing
  - Parallel processing