Caching in Video CDNs: Building Strong Lines of Defense

Kianoosh Mokhtarian, Hans-Arno Jacobsen

EuroSys Conference
April 2014
How does a typical CDN look?
How does a typical CDN look?
Content management

(Logically) centralized
Distributed: Cooperative caching

Why not?
Scale
Dynamics
CDN concerns & objectives
Each server maintains the most locally popular videos w.r.t. the request traffic it receives

HTTP 302 for unpopular ones
Challenges

1. Cache fill or redirect?
   Inherent tradeoff

![Graph showing the relationship between Ingress (%) and Redirect (%) with a downward trend.]
Challenges

2. Different fill-vs-redirect preferences
   i.e. different sweetspots

   Uplink cost/constraints
   Egress utilization
   Overloaded disk

![Graph showing relationship between Ingress and Redirect]
Challenges

3. Diverse intra-file popularities
   Arbitrary byte-range requests
Contributions

xLRU Cache (baseline)

Cafe Cache (ingress-efficient)

Psychic Cache (offline; greedy)

Optimal Cache (offline; limited scale)

Evaluated with actual data from a global CDN
The VCDN caching problem

Request = \{timestamp, videoID, byte range\}

Servered/redirected in full

File Chunks of fixed $K$ bytes

$\Rightarrow$ Unit of stored data = \{videoID, chunkID\}

Fill-vs-redirect config

$C_F, C_R$: Cost of Fill and Redirect; $\alpha = \frac{C_F}{C_R}$

\[\alpha > 1 \text{ (ingress constrained)}\]
\[\alpha = 1\]
\[\alpha < 1\]
Cache efficiency

**Total cost** = ingress bytes $\times C_F +$ redirect bytes $\times C_R$

**Cache efficiency** for $\alpha = 1$ ($C_F = C_R$):

$$1 - \frac{\text{everything not cache hit}}{\text{all requests}} =$$

$$1 - \frac{\text{bytes served by fill} + \text{bytes redirected}}{\text{total requested bytes}}$$

**General-case efficiency:**

$$1 - \frac{\text{bytes served by fill} \times C_F + \text{bytes redirected} \times C_R}{\text{total requested bytes}}$$

$C_F + C_R = 2$

$C_F / C_R = \alpha$
Baseline solution: xLRU

Request: \{videoID, chunk range\}

- Not popular enough => Redirect
- Serve

Popularity Manager
File level
LRU admission

\( DT_v = \) Time since last request for video
\( DT_0 = \) Time since last request for oldest chunk on disk

\( DT_v < DT_0 \)

Yes (serve)

Evict 0 or more Least Recently Used chunks
Cache-fill the requested (but missing) chunks

Disk Cache: Chunk-level, LRU replacement
Baseline solution: xLRU

Request: \{videoID, chunk range\}

Not popular enough => Redirect

\( DT_v = \text{Time since last request for video} \)
\( DT_0 = \text{Time since last request for oldest chunk on disk} \)

\( DT_v \times \left( \frac{C_F}{C_R} \right) < DT_0 \)

Yes (serve)

Serve

Evict 0 or more Least Recently Used chunks
Cache-fill the requested (but missing) chunks

Popularity Manager
File level
LRU admission

Disk Cache: Chunk-level, LRU replacement
Chunk-Aware Fill-Efficient (CAFE)

Estimate the joint cost of the current as well as the expected future requests

\[ E[\text{Cost}_{\text{serve}}] \text{ vs } E[\text{Cost}_{\text{redirect}}] \]

Based on \( C_F, C_R \)

EWMA-ed inter-arrival times

Availability of chunks

Serve or redirect accordingly
Optimal cache (offline)

Input:

Full sequence of (future) requests

Output for each request:

Serve or redirect?
What to evict?

Requests:

#1: video A, 2--5: 2 3 4 5
#2: video B, 1--2: 1 2
#3: video A, 4--6: 4 5 6
...

Server's disk capacity:
Psychic cache (offline; greedy)

Greedy (scalable) replacement for Optimal Cache

Chunks scored based on future, not historic requests
  Admission
  Replacement

Final decision made similarly to Cafe
# Summary, so far ...

<table>
<thead>
<tr>
<th><strong>xLRU</strong> Cache</th>
<th><strong>Optimal</strong> Cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>O(1) operations</td>
<td>Offline</td>
</tr>
<tr>
<td>Based on LRU policies</td>
<td>LP relaxation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Cafe</strong> Cache</th>
<th><strong>Psychic</strong> Cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>E[Cost\textsubscript{serve}] vs E[Cost\textsubscript{redirect}]</td>
<td>Offline</td>
</tr>
<tr>
<td>O(1) &amp; O(log N) operations</td>
<td>Greedy heuristic</td>
</tr>
<tr>
<td></td>
<td>E[Cost\textsubscript{serve}] vs E[Cost\textsubscript{redirect}]</td>
</tr>
</tbody>
</table>
Evaluation: Setup

Actual CDN traces

1-month period in 2013
6 selected servers across all continents

Entries (anonymized):

\{Timestamp, video ID, byte range\}
XLRU, Cafe and Psychic

The server in Europe

$\alpha = 2$

Cafe & Psychic more accurate in approving cache-fill for the right content

Cafe: 10% increase in cache efficiency from xLRU

Psychic (i.e. max): 13%
Different operating points vs $\alpha$
Different disk sizes ($\alpha = 2$)
On different servers
Take away

**xLRU Cache**
- Lightweight
- Best suited for non-constrained ingress

**Psychic Cache**
- Offline (knows future)
- Greedy
- For benchmarking

**Cafe Cache**
- High cache efficiency
- Accurate control of fill-vs-redirect

**Optimal Cache**
- Limited scale
- Shows Psychic is a reasonable upperbound indicator
Optimal cache (offline)

Linear programming relaxation

\( X[i, t] \): whether object \( i \) is in cache at time \( t \)

\( A[i] \): whether to serve or redirect request \( i \)

Objective

\[ \text{min} \ [\text{filled bytes} \times C_F + \text{redirected bytes} \times C_R] \]

Constraints

Disk size

Request wholeness

Inter-LP-variable consistencies ...
Psychic: An estimator of max expected efficiency

Optimal Cache in limited scale

- 2 days; 100 selected files; 20 MB file size cap
- Disk = 5% of all requested chunks

No xLRU and Cafe yet
Related work

Cooperative caching

CDN-wide content placement

Classic cache replacement