An Analysis of Facebook Photo Caching

Qi Huang, Ken Birman, Robbert van Renesse, Wyatt Lloyd, Sanjeev Kumar, Harry C. Li
The problem

- 250 billion photos on Facebook in 2013
- 350 million more uploaded daily
- High availability and low latency required for delivery infrastructure.
Facebook's Photo-Serving Stack

Client
- Browser cache

Facebook storage and caches

Akamai CDN
Facebook's Photo-Serving Stack

Focus exclusively on Facebook stack
Edge Caches (FIFO)

- tens of independent edge caches across the US
Edge Caches (FIFO)

Purpose:

- reduce latency
- reduce bandwidth utilization between clients and datacenters
Global Origin Cache (FIFO)

- a single global cache
- requests routed to specific data centers based on hash of photo id
Global Origin Cache (FIFO)

Purpose:

- traffic sheltering for I/O bound backend
Haystack backend

- located in the same data centers as origin servers
Resizers

- photos are served in different sizes to different users
- resized photos are treated as distinct objects by the cache layer
Methodology
Stack instrumentation

- desktop clients only
- data for browser cache inferred by counting requests seen at browsers and edge caches
Sampling method

- request-based sampling: collect X% of requests
- object-based sampling: collect X% of objects
Sampling method

- request-based sampling: collect X% of requests
- object-based sampling: collect X% of objects
  - by deterministic test on photo id
Object-based Sampling

- better coverage of unpopular photos
- simplified cross-stack analysis
Analysis
The collected data

- One-month long trace
- 77M requests from 13.2M user browsers
- 1.3M unique photos (2.6M photo objects)
Popularity Distribution

- approximately Zipfian
Traffic share by photo popularity

- Browser
- Edge
- Origin
- Backend

Percent of requests

High | Medium | Low | Lowest
--- | --- | --- | ---
0 | 0 | 0 | 0
Geographic Traffic Distribution
Client To Edge Cache Traffic
Client To Edge Cache Traffic

- Atlanta: 20%
- California: 5%
- New York: 5%
- Washington D.C.: 35%
- Dallas: 10%
- Miami: 10%
Client To Edge Cache Traffic

- routing algorithm considers latency, edge capacity and ISP peering cost
- substantial remote traffic is normal
Edge Cache to Origin Cache Traffic

Traffic flow based on content, not locality.
Cross-Region Traffic at Backend

- Caused by data migrations and hardware failures
- Only 0.2% of Origin Cache to Backend traffic

<table>
<thead>
<tr>
<th>Origin Cache Server Region</th>
<th>Backend Region</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Virginia</td>
<td>North Carolina</td>
<td>Oregon</td>
</tr>
<tr>
<td>Virginia</td>
<td>99.885%</td>
<td>0.049%</td>
<td>0.066%</td>
</tr>
<tr>
<td>North Carolina</td>
<td>0.337%</td>
<td>99.645%</td>
<td>0.018%</td>
</tr>
<tr>
<td>Oregon</td>
<td>0.149%</td>
<td>0.013%</td>
<td>99.838%</td>
</tr>
<tr>
<td>California</td>
<td>24.760%</td>
<td>13.778%</td>
<td>61.462%</td>
</tr>
</tbody>
</table>
Potential Improvements
Cache Simulation

- replay the trace
- use the first 25% to warm the cache
- evaluate using the remaining 75% of the trace
Object-hit Ratios for Edge Cache

- S4LRU improves 9% at $x$ over FIFO
- Still a lot of room for improvement
- Four queues are maintained at levels 0 to 3
● Missed objects inserted into the head of level 0 queue
S4LRU

- On a cache hit, the item is moved to the head of the next higher queue
S4LRU

- items evicted from the tail of a queue to the head of the next lower queue
Byte-hit Ratios for Edge Cache

- LFU fails to reduce bandwidth utilization over the current solution
Origin cache

S4LRU improves Origin slightly more than Edge (14%)
Social-Network Analysis
Content Age Effect
Traffic share for non-profile photos by age
Traffic share for social activity groups.