CSC2231: Caching + Zipf

http://www.cs.toronto.edu/~stefan/courses/csc2231/05au

Stefan Saroiu
Department of Computer Science
University of Toronto
Administrivia

- No lecture on Monday because of Cascon
- Research reports due next Wednesday
  - In less than 1 week!!
Cache Hit Rates

• **Two ways to measure cache hit rates:**
  – Object hit rate
    • Reduces latency
    • Reflects caching benefits to users
  – Byte hit rate -- reduced bandwidth
    • Reduces bandwidth
    • Reflects caching benefits to network

• **Typically for the Web:**
  – Byte HR < Object HR
  – What does this mean?
What drives cache hit rates?
What drives cache hit rates?

- Popularity distribution
- Number of clients
- Rate of updates to the documents
- Cacheability of data
- Cache sizes vs. object sizes
Web popularity distribution
Artificial Zipf distribution with $\alpha=0.8$
0.01% of documents
8% of all accesses
0.01% of documents
8% of all accesses

0.1% of documents
18% of all accesses
The graph illustrates the 80/20 rule for document access. It shows that 0.01% of documents account for 8% of all accesses, 0.1% of documents account for 18% of all accesses, and 10% of documents account for 60% of all accesses. This distribution is represented on a logarithmic scale, with the x-axis showing the documents ranked and the y-axis showing the number of accesses. The line on the graph represents this distribution, highlighting the disproportionate access patterns.
90% of documents:
• have at most 6 accesses
• account for 40% of all accesses
Implications of Object Popularity

- **Implications of object popularity to cache hit rates**
  - Lots of unpopular objects --- don’t cache
  - Significant very popular objects --- cache
  - Grey area --- ?
What drives cache hit rates?

- Popularity distribution
- Number of clients
- Rate of updates to the documents
- Cacheability of data
- Cache sizes vs. object sizes
Hit Rate vs. Population
Implications of Client Population

- **What are the implications of client population size to caching hit rates?**
  - Cache location:
    - Trade-off:
      - Closer to the user, higher benefit
      - Closer to the user, fewer clients
    - Place it at the “sweet-spot”
  - Cache hierarchies:
    - Hit rate grows slowly above a few 1000 users
    - Each layer adds latency (and bandwidth)
What drives cache hit rates?

- Popularity distribution
- Number of clients
- Rate of updates to the documents
- Cacheability of data
- Cache sizes vs. object sizes
Rate of Updates

• **Old studies ‘94 - ‘00**
  – Average lifetime of an HTML object: 40-50 days
  – Average lifetime of an image: ~100 days
  – More popular the object, the more often it is updated
  – Hard to predict
    • More popular the object, easier it is to predict

• **Implications:**
  – Content expiration:
    • Hard to get right; unclear if worth doing it
What drives cache hit rates?

- Popularity distribution
- Number of clients
- Rate of updates to the documents
- Cacheability of data
- Cache sizes vs. object sizes
Active-Caching

• Java CacheApplet cached with each object
• On each request, cache invokes applet to:
  – Generate reply, use cached copy, trigger revalidation
  – Maintains on-cache persistent state

• Problems in practice:
  – Cache monitors applet CPU and storage use
    • Can “evict” applet and revert to “Expires” consistency
  – Does this solve any real problems?
    • Advertising? Commerce? Personalization?
Advertising: Cache-Busting

- **Advertisers want to** track and target
  - Per-user cookies
  - Per-user, per-pageview ad selection
  - Caches **defeat** these goals
  - Caches **help** deliver ad content quickly

- **Real-world solution:**
  - Redirect for ad selection/logging
  - Cacheable ad image files
Commerce Databases

Selling

• Product DB
  – Inventory, descriptions, promotions, $, cross-selling info, …

• User DB
  – Purchase history, recent browsing, …

• Business rules

Purchasing

• User DB
  – Credit card, shipping address, …

• Transaction system
  – Credit card clearance, integration to shipping, …

Distributed databases for this? Privacy, proprietary concerns?
Advertising Databases

• **Ad information:**
  – “Inventory”, $, targeting criteria (eligible content types, desired user types, time of day), …

• **Placement information:**
  – Stats about the traffic to different pages, $, content topic, expected mix of users with different criteria, …

• **Per-user information:**
  – Topics of interest, links to registration/marketing profiles, detailed recent ad viewing history, …

• **Business rules for combining the above in real-time**

• **How to distribute these databases to caches?**
Personalized Publishing

• my.yahoo.com, slashdot.org
  – Personalized pages from sharable/cacheable components
  – Different layouts/subsets/orders sorts

• Seems more tractable
  – Per-user preferences database must be distributed

• Active-caching? Other cache-side method? Better done at the client w/XML?
Delta-Coding

- **Server sends “diffs” against cached copy of page**
  - Can reduce bandwidth for dynamic content
  - Still requires round-trip latency to server
    - For most WWW objects probably not worth saving bw

- **Exploits redundant data already in cache to compress updated object**
  - Straight compression (orthogonal)
  - Spring/Wetherall use of Manber fingerprints?
Dynamic Content vs Cache Deployments

<table>
<thead>
<tr>
<th></th>
<th>Dynamic: Advertising</th>
<th>Dynamic: Commerce</th>
<th>Dynamic: Publishing</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client Cache</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proxy Cache</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CDN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accelerator</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Dynamic Content vs Cache Deployments

<table>
<thead>
<tr>
<th></th>
<th>Dynamic: Advertising</th>
<th>Dynamic: Commerce</th>
<th>Dynamic: Publishing</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Client Cache</strong></td>
<td>Delta-coding</td>
<td>Delta-coding</td>
<td>Delta-coding</td>
<td>Delta-coding</td>
</tr>
<tr>
<td><strong>Proxy Cache</strong></td>
<td>Delta-coding</td>
<td>Delta-coding</td>
<td>Active-caching, Delta-coding</td>
<td>Delta-coding</td>
</tr>
<tr>
<td><strong>CDN</strong></td>
<td>Custom (DoubleClick)</td>
<td>Custom (Amazon?, Yahoo Merchants)</td>
<td>Active-caching, Delta-coding, Custom (Akamai)</td>
<td>Custom (Akamai)</td>
</tr>
<tr>
<td><strong>Accelerator</strong></td>
<td>Webserver / DUP</td>
<td>Webserver / DUP</td>
<td>Webserver / DUP</td>
<td>Expires</td>
</tr>
</tbody>
</table>
What drives cache hit rates?

- Popularity distribution
- Number of clients
- Rate of updates to the documents
- Cacheability of data
- **Cache sizes vs. object sizes**
Cache Sizes

- Google can cache the entire Internet
- Disks are infinite

- Do we care about Web cache sizes anymore?
Discussion

• RSS feeds:
  – Should we cache these?