Web Caching

Robert Grimm
New York University
Before We Get Started

- Illustrating Results
- Type Theory 101
Illustrating Results

Average response time

Dropped connections

1-6 correspond to 10/20/50/100/100/150 concurrent clients where 5 includes 25 “faulty” clients
What is a type?
- “Qualities common to a number of individuals that distinguish them as an identifiable class” [Merriam-Webster]

Why do we care?
- Help us reason about the meaning of programs

How can we do this formally?
- One approach: rewrite rules
  - Axioms (e.g., “() matches ()”)
  - Inference rules
    - Value matches Type1
      -----------------------------
      Value matches Type1 | Type2
Web Caching
What’s in a Model?

- Some mathematical formulation about reality
- Why do we care?
  - Predict the future
  - Evaluate algorithms
    - Effectiveness
    - Limitations
  - Project systems behavior
    - Very large client populations
- What’s hard about models?
  - Identifying a model
  - Verifying a model
Breslau et al. Reality

- Six web proxy traces
  - Digital Equipment (nee Compaq nee HP)
  - University of California at Berkeley (Home IP service)
  - Questnet (Australian ISP)
  - National Lab for Applied Networking Research
  - FuNet (academic ISP in Finland)
Breslau et al. Analysis

0.77

0.69

0.78

0.73

0.64

0.83
Breslau et al.
Observations

- Request distribution is indeed Zipf-like
- “10/90” rule does not hold
  - 25-40% of documents draw 70% of web accesses
- Low statistical correlation between
  - Document access frequency
  - Document size
- Hardly any statistical correlation between
  - Document access frequency
  - Document update rate
Breslau et al. Model

- Stream of requests for $N$ web pages, ranked by popularity
- Probability request is for page $I$

$$P_N(i) = \frac{\Omega}{i^\alpha} \quad \text{where} \quad \Omega = \left( \sum_{i=1}^{N} \frac{1}{i^\alpha} \right)^{-1}$$

- Each request is independent from others
- No cache invalidations
Breslau et al. Implications

- Hit ratio grows logarithmically or like a small power with number of requests
  - Consistent with data, other researchers’ observations
- Independent reference model suggests least-frequently-used cache replacement policy
  - But, GD-Size performs better for small cache sizes and LRU has decent byte hit ratios
    - What about temporal effects?
Cooperative Caching

- Basic idea
  - Several caches work together to provide a larger cache
- Why do we care?
  - We hope that a larger cache gives us better hit rates
- Possible organizations
  - Hierarchical
  - Hash-based
  - Directory-based
What is the best performance one could achieve with “perfect” cooperative caching?

For what range of client populations can cooperative caching work effectively?

Does the way in which clients are assigned to caches matter?

What cache hit rates are necessary to achieve worthwhile decreases in document access latency?
Wolman et al. Traces

- From University of Washington and Microsoft

<table>
<thead>
<tr>
<th>Parameter</th>
<th>UW</th>
<th>Microsoft</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Requests</td>
<td>82.8 million</td>
<td>107.7 million</td>
</tr>
<tr>
<td>HTTP Objects</td>
<td>18.4 million</td>
<td>15.3 million</td>
</tr>
<tr>
<td>Total Bytes</td>
<td>677 GB</td>
<td>(N/A)</td>
</tr>
<tr>
<td>Ave. Requests/s</td>
<td>137</td>
<td>199</td>
</tr>
<tr>
<td>Clients</td>
<td>22,984</td>
<td>60,233</td>
</tr>
<tr>
<td>Servers</td>
<td>244,211</td>
<td>360,586</td>
</tr>
<tr>
<td>Duration</td>
<td>7 days</td>
<td>6 days 6 hours</td>
</tr>
</tbody>
</table>
Infinite-size caches
  - No capacity misses, but compulsory misses

Two types of caches
  - Ideal
    - Everything is cacheable
  - Practical
    - HTTP/1.1 cache control headers, no-cache pragmas
    - Cookies
    - Object names with suffixes mapping dynamic objects
    - Uncacheable methods
    - Authorization, Vary header fields
Wolman et al.
Hit Rate vs. Population

- Why is Microsoft’s ideal rate higher than UW’s?
- How many caches should we deploy?
What is the impact of population size on latency?
Wolman et al. How to Save Bandwidth

- How do shared objects compare to other objects in size?
- How does population size impact bandwidth consumption?
Wolman et al.
Hit Rate vs. Organizations

- What is the effect of organizations?
  - Real
  - Random
  - Interest-based

- What is the effect of cooperative caching between organizations?
What is the correlation between sharing and cacheability?

Are the population limits?
Wolman et al.
Hit Rate vs. Cooperation

- What is the degree of sharing between organizations?
- What is the case for unpopular documents?
Just like Breslau et al., **but**

- Steady-state performance rather than finite sequence
  - What is the relationship between
    - Rate of change of documents
    - Rate of addition of documents
    - Rate of requests for documents
    - Size of population

- Incorporates document rate of change
  - Exponential distribution
  - Independent of document size and latency
  - Dependent on popularity
    - What’s the intuition here?
**Wolman et al.**  
**Rate of Change**

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Popular Mean</th>
<th>Popular Median</th>
<th>Unpopular Mean</th>
<th>Unpopular Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>14</td>
<td>&lt;1</td>
<td>186</td>
<td>85</td>
</tr>
<tr>
<td>Always change</td>
<td>3</td>
<td>&lt;1</td>
<td>129</td>
<td>23</td>
</tr>
<tr>
<td>Never change</td>
<td>27</td>
<td>&lt;1</td>
<td>763</td>
<td>180</td>
</tr>
<tr>
<td>Cacheable</td>
<td>5</td>
<td>&lt;1</td>
<td>168</td>
<td>65</td>
</tr>
<tr>
<td>Uncacheable</td>
<td>&lt;1</td>
<td>&lt;1</td>
<td>22</td>
<td>&lt;1</td>
</tr>
</tbody>
</table>
What is the impact of rate of change on hit rate?
Again, what is the impact of rate of change on hit rate?
Wolman et al.
Cooperative Caching

<table>
<thead>
<tr>
<th>City</th>
<th>Hierarchical</th>
<th>Hashed</th>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrival Rate</td>
<td>$r_1 = 150M/day$</td>
<td>$30M/day$</td>
<td>$37M/day$</td>
</tr>
<tr>
<td></td>
<td>$r_2 = 30M/day$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latency</td>
<td>0.86 secs</td>
<td>0.88 secs</td>
<td>0.89 secs</td>
</tr>
<tr>
<td>Storage</td>
<td>11 TB</td>
<td>1.5 TB</td>
<td>9.5 TB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State</th>
<th>Hierarchical</th>
<th>Hashed</th>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrival Rate</td>
<td>$r_1 = 1.37B/day$</td>
<td>29.5M/day</td>
<td>37.7M/day</td>
</tr>
<tr>
<td></td>
<td>$r_2 = 147M/day$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$r_3 = 29.5M/day$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latency</td>
<td>0.79 secs</td>
<td>0.83 secs</td>
<td>0.85 secs</td>
</tr>
<tr>
<td>Storage</td>
<td>11 TB</td>
<td>150 GB</td>
<td>9.5 TB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>West Coast</th>
<th>Hierarchical</th>
<th>Hashed</th>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrival Rate</td>
<td>$r_1 = 13B/day$</td>
<td>29.5M/day</td>
<td>37.9M/day</td>
</tr>
<tr>
<td></td>
<td>$r_2 = 1.37B/day$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$r_3 = 147M/day$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$r_4 = 29.5M/day$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latency</td>
<td>0.78 secs</td>
<td>1.1 secs</td>
<td>1.13 secs</td>
</tr>
<tr>
<td>Storage</td>
<td>11 TB</td>
<td>15 GB</td>
<td>9.5 TB</td>
</tr>
</tbody>
</table>

- What about latency? Request rate?
Little need for more work on cooperative caching
Largest benefit achieved with small populations
Performance limited by cacheability
Mutual interest does not provide advantages

What about the effects of
  Dynamic documents?
  Streaming multimedia?
What about protocols?