Web Caching and Content Distribution: A View From the Interior

Syam Gadde
Jeff Chase
Duke University

Michael Rabinovich
AT&T Labs - Research
Overview

- Analytical tools have evolved to predict behavior of large-scale Web caches.
  - Are results from existing large-scale caches consistent with the predictions?
    - NLANR
  - What do the models predict for Content Distribution/Delivery Networks (CDNs)?
- **Goal**: answer these questions by extending models to predict *interior* cache behavior.
Generalized Cache/CDN (External View)

Origin Servers

CDNs

Web Caches

Clients

{push, request, reply}

{request, reply}
Generalized Cache/CDN (Internal View)

Request Routing Function $f$

Interior Caches
- root caches
- reverse proxies

Leaf Caches
- bound client populations
Goals and Limitations

- Focus on *interior cache* behavior.
  - Assume leaf caches are ubiquitous.
  - Model CDNs as interior caches.
- Focus on *hit ratio* (percentage of accesses absorbed by the “cloud”).
  - Ignore push replication; at best it merely reduces some latencies by moving data earlier.
- Focus on “typical” *static* Web objects.
  - Ignore streaming media and dynamic content.
Outline

- Analytical model
  - applied to interior nodes of cache hierarchies
  - applied to CDNs
- Implications of the model for CDNs in the presence of ubiquitous leaf caching
- Match model with observations from the NLANR cache hierarchy
- Conclusion
Analytical Model

- [Wolman/Voelker/Levy et. al., SOSP 1999]
  - refines [Breslau/Cao et. al., 1999], and others
- Approximates asymptotic cache behavior assuming Zipf-like object popularity
  - caches have sufficient capacity
- Parameters:
  - $\lambda = \text{per-client request rate}$
  - $\mu = \text{rate of object change}$
  - $p_c = \text{percentage of objects that are cacheable}$
  - $\alpha = \text{Zipf parameter (object popularity)}$
Cacheable Hit Ratio: the Formula

- $C_N$ is the hit ratio for cacheable objects achievable by population of size $N$ with a universe of $n$ objects.

\[
C_N = \int_1^n \frac{1}{Cx^\alpha} \left( \frac{1}{1 + \frac{\mu Cx^\alpha}{\lambda N}} \right) dx
\]

\[
C = \int_1^n \frac{1}{x^\alpha} dx
\]

[Wolman/Voelker/Levy et. al., SOSP 99]
Inside the Hit Ratio Formula

Approximates a sum over a universe of $n$ objects...

...of the probability of access to each object $x$...

...times the probability $x$ was accessed since its last change.

$C_N = \int_1^n \frac{1}{C x^\alpha} \frac{1}{1 + \frac{\mu C x^\alpha}{\lambda N}} dx$

$C$ is just a normalizing constant for the Zipf-like popularity distribution (a PDF).

$C = \frac{1}{\Omega}$

in [Breslau/Cao 99]

$0 < \alpha < 1$
An Idealized Hierarchy

Assume the trees are symmetric to simplify the math.
Ignore individual caches and solve for each level.
Hit Ratio at Interior Level $i$

- $C_N$ gives us the hit ratio for a complete subtree covering population $N$
- The hit ratio predicted at level $i$ or at any cache in level $i$ is given by:

$$\frac{\text{hits at level } i}{\text{requests to level } i} = h_i = \frac{R\rho_c (C_{N_i} - C_{N_{i+1}})}{r_i} = \frac{r_{i+1} - h_{i+1}}{r_{i+1} - h_{i+1}}$$

"the hits for $N_i$ (at level $i$) minus the hits captured by level $i+1$, over the miss stream from level $i+1"
Predicted hit ratio for cacheable objects, observed at root of a two-level cache hierarchy (i.e. where $r_2 = R_p c$):

$$
\frac{h_1}{r_1} = \frac{C_{N_1} - C_{N_2}}{1 - C_{N_2}}
$$
Generalizing to CDNs

Symmetry assumption: $f$ is stable and “balanced”.

$f(leaf, object, state)$
Servers

Interior Caches

Leaf Caches

$N_1$ clients

$N_1$ clients
What happens to $C_N$ if we partition the object universe?
Hit ratio in CDN caches

Given the symmetry and balance assumptions, the cacheable hit ratio at the interior (CDN) nodes is:

\[
\frac{C_{N_I} - C_{N_L}}{1 - C_{N_L}}
\]

\(N_I\) is the covered population at each CDN cache. \(N_L\) is the population at each leaf cache.
We apply the model to gain insight into interior cache behavior with:

- varying leaf cache populations ($N_L$)
  - e.g., bigger leaf caches
- varying ratio of interior to leaf cache populations ($N_I/N_L$)
  - e.g., more specialized interior caches
- Zipf $\alpha$ parameter changes
  - e.g., more concentrated popularity
Analysis (cont’d)

- Fixed parameters (unless noted otherwise):
  - $\lambda$ (client request rate) = 590 reqs./day
  - $\mu$ (rate of object change) =
    - once every 14 days (popular objects, 0.3%)
    - once every 186 days (unpopular objects)
  - $p_c$ (percent of requests cacheable) = 60%
  - $\alpha$ (Zipf parameter - object popularity) = 0.8
Cacheable interior hit ratio observed at interior level fixing interior/leaf population ratio

Increasing $N_I$ and $N_L$ -->
Interior hit ratio
as percentage of all cacheable requests, fixing interior/leaf population ratio

increasing $N_I$ and $N_L$ -->

marginal cacheable hit ratio
Cacheable interior hit ratio
fixing leaf population

cacheable
hit
ratio

increasing “bushiness” -->
Cacheable interior hit ratio as percentage of all requests fixing leaf population

marginal cacheable hit ratio

increasing “bushiness” -->
Cacheable interior hit ratio as percentage of all requests varying Zipf $\alpha$ parameter

$N_L$ fixed at 1024 clients
Cacheable interior hit ratio as percentage of all requests varying Zipf $\alpha$ parameter
Conclusions (I)

- Interior hit ratio captures effectiveness of upstream caches at reducing access traffic filtered by leaf/edge caches.
  - Hit ratios grow rapidly with covered population.
- Edge cache populations ($N_L$) are key: is it one thousand or one million?
  - With large $N_L$, interior ratios are deceptive.
  - At $N_L = 10^5$, interior hit ratios might be 90%, but the CDN sees less than 20% of the requests.
Correlating with NLANR Observations

- Do the predictions match observations from existing large-scale caches?
- Observations made from traces provided by NLANR (10/12/99).
  - Observed total hit ratio at (unified) root is 32%
  - 200 of the 914 leaf caches in the trace account for 95% of requests
  - Daily request rate indicates population is on the order of tens of thousands
  - What is the predicted N?
Model vs. Reality

- NLANR roots cooperate; we filter the traces to determine the *unified* root hit ratio.
- NLANR caches are bounded; traces imply that capacity misses are low at 16GB.
- Analysis assumes the population is balanced across the 200 leaves of consequence.
- Analysis must compensate for objects determined to be uncacheable at a leaf.
Cacheable interior hit ratio
varying percentage of requests detected as uncacheable by leaves

Cacheable hit ratio

200+ leaf caches
Cacheable interior hit ratio 
varying percentage of requests detected as 
uncacheable at request time

1000 clients per leaf cache
Conclusions (II)

- NLANR root effectiveness is around 32% today; it is serving its users well.
- NLANR experiment could validate the model, but more data from the experiment is needed.
  - E.g., covered populations, leaf summaries
- The model suggests that the population covered by NLANR is relatively small.
- With larger N and N_L, higher root hit ratios are expected, with lower marginal benefit.
Modeling CDNs

- If the routing function satisfies three properties:
  - an interior cache sees all requests for each assigned object $x$ from a population of size $N_i$
  - every interior cache sees an equivalent object popularity distribution ($n/\lambda$ held constant)
  - all requests are routed through leaf caches that serve $N_L$ clients

- Then interior cacheable hit ratio is:

$$\frac{C_{N_i} - C_{N_L}}{1 - C_{N_L}}$$
Hit ratio with detected uncacheable documents

- $p_u$ is the percentage of uncacheable requests detected at request time (and not forwarded to parents):

\[
\frac{h_i}{r_i} = \frac{R p_c (C_{N_i} - C_{N_{i+1}})}{R - h_{i+1} - (1 - p_c)(1 - p_u)r_{i+1}}
\]

\[
\frac{h_1}{r_2} = \frac{H_{N_1} - H_{N_2}}{1 - H_{N_2} - (1 - p_c)(1 - p_u)}
\]
Cache Hierarchies

- As introduced by the Harvest project
  - $k$ levels of demand-side caches arranged in a tree (for now)
  - clients are bound to leaves
  - each node’s miss stream routes to its parent
- As extended by NLANR (Squid)
  - NLANR-operated root caches cooperate by partitioning URL space
Cache Hierarchies Illustrated