# Variable-QoS Resource Allocation: Lessons Learned from Web Caching

Terence Kelly, Sugih Jamin
Electrical Engineering & Computer Science

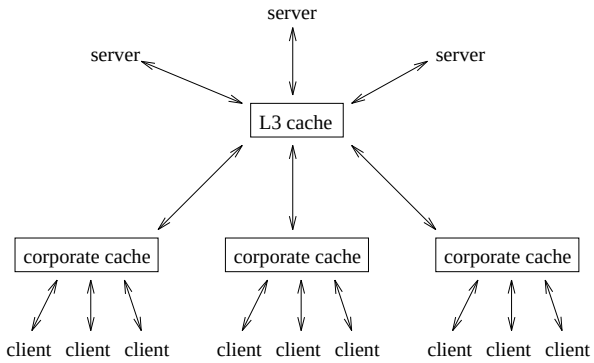
Jeffrey K. MacKie-Mason
Economics & School of Information

University of Michigan
Ann Arbor MI 48109-2110

2 December 1999

$Id: modd.tex,v 1.37 1999/11/18 03:34:53 tpkelly Exp $

# Web Caching 101




Easy: static content is read-only, so fewer worries about cache coherence, need not write back modified pages.

Benefits: reduce network traffic, server load, client latency.

Important because Web accounts for most important traffic on the Internet, and largest volume.

# Variable-QoS Web Caching

Caching provides way to trade disk space for bandwidth, server load, end-user patience.

Furthermore shared caches are obvious loci for variable-QoS mechanisms because scarce disk space can be diverted to serve those who value caching most.

Pay-for-cache schemes are commercially viable (e.g., Akamai), however variable-QoS mechanisms to ensure that caches deliver highest possible value to users are not present.

Our goal: allocate cache space to increase aggregate user utility.

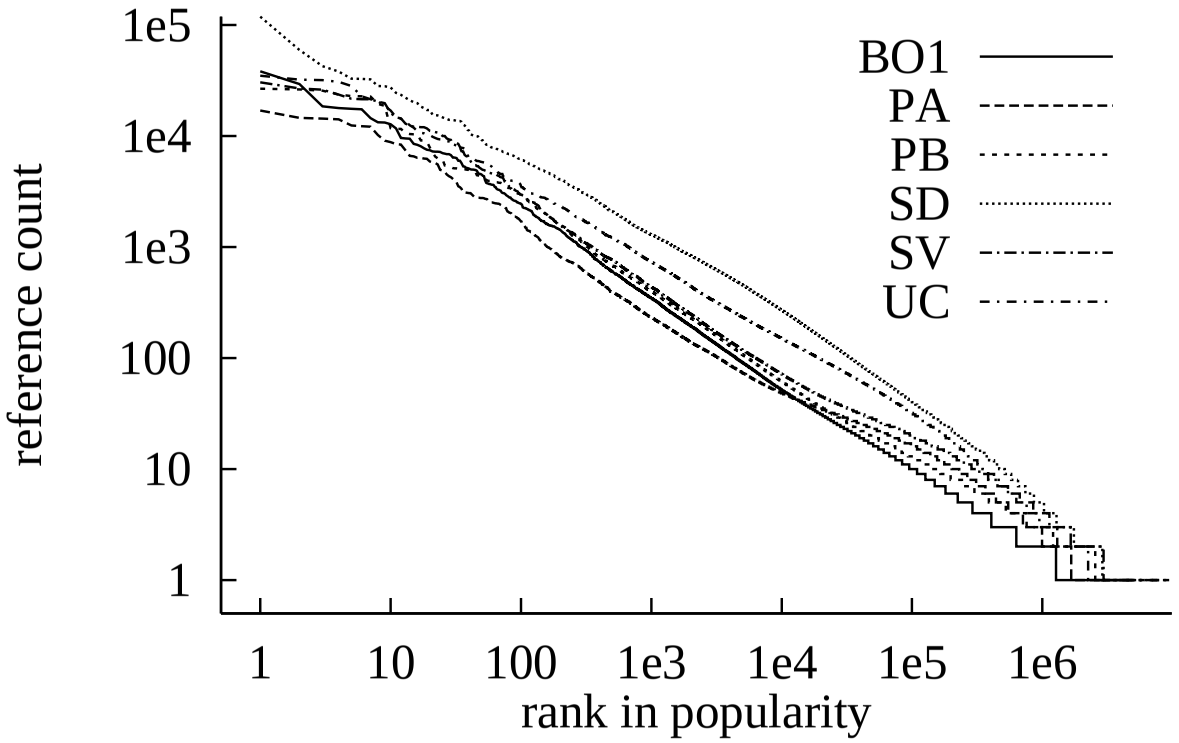
# Web Cache Workload

Measured characteristics of L3 Web request streams:

- Low temporal locality of reference—consecutive requests for same document separated by many requests for *other* documents: no reason to prefer LRU
- Zipf-like document popularity distribution
- Non-stationary popularity distribution

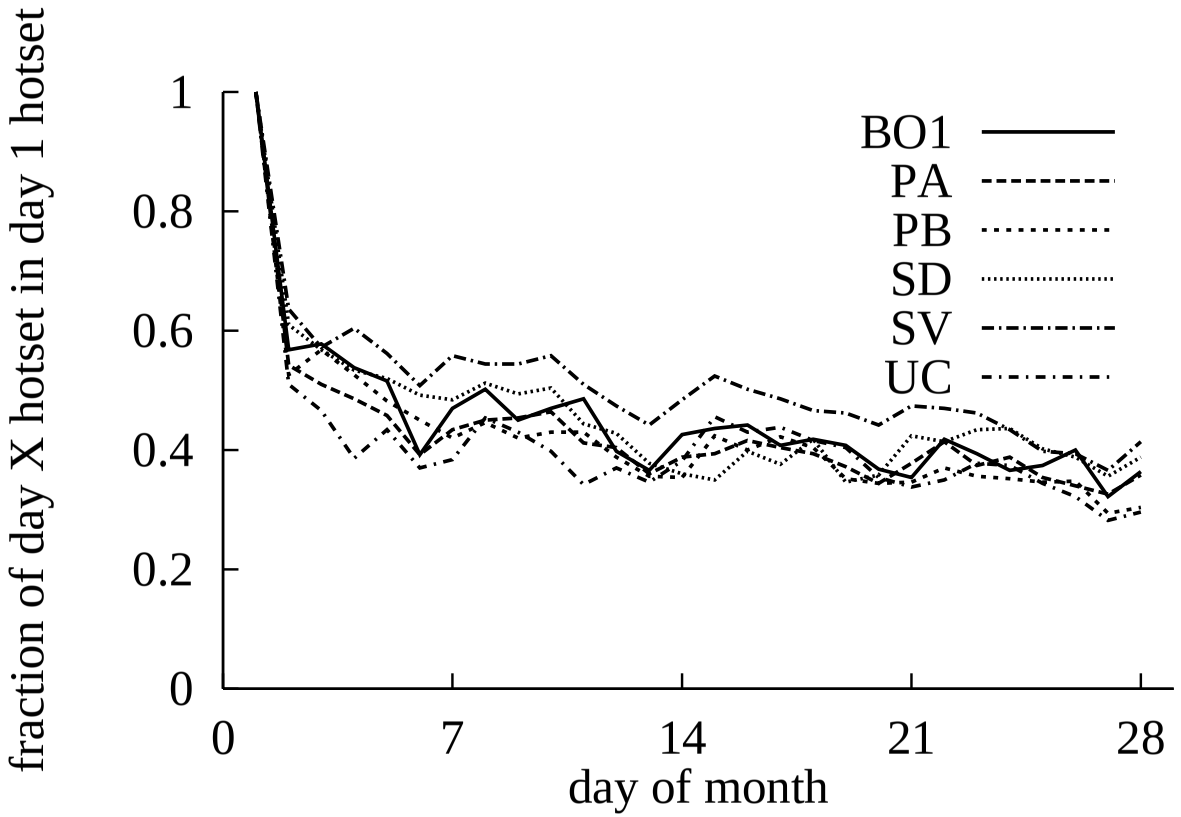
*User preferences* are part of our workload, and at present we have no data on Web stakeholder preferences. Indirect estimates from bandwidth purchasing decisions (commercial and experimental) might provide useful approximations for future research. Direct data might become available from commercial caching firms.

# Zipf-like popularity distribution




Implication: prefer LFU-like algorithms

# “Hot set drift”




Implication: aging mechanism needed to prevent cache pollution

# Two Approaches

This talk: Value-sensitive replacement policy (Kelly et al.)

- Weighted LFU with aging
- Generalizes conventional replacement policies (LFU, LRU)
- Well suited to cache dynamics
- Performs reasonably well at all cache sizes
- *Not* incentive compatible, but not perverse either

Another way: Periodic cache space auctions (Chan et al.)

- Incentive-compatible: Distributes storage of reference counts back to servers, deters false reporting
- With tested parameters, worked well only at extremely small cache sizes

# Weighted LFU with Aging

Remove documents in ascending order of weighted reference count: $W_u \times N_u$ where $N_u$ is number of references to document $u$ and $W_u$ indicates extent to which $u$'s origin server values caching. Optimal if document popularity distribution is stationary Zipf.

Tunable aging mechanism: Every $K$th eviction, select least-recently-referenced (LRU) document as victim. Needed because document popularity distribution changes over time. $K$ parameter must be tuned.

Performs well on real request patterns, *some* $W_u$ distributions.

Other biased replacement policies exist: "Greedy-Dual Size", GDSF, "Lowest Unified Value."

# Unexpected Problems

Zipf distribution: Document reference counts $N_u$ span five orders of magnitude in our traces. If weights $W_u$ span a much narrower range, e.g., one order of magnitude, differences in document popularity swamp differences in preferences.

If *client* preferences rather than server preferences are used to bias the allocation of disk space, and if client preferences are uncorrelated with reference counts, conflicting client preferences tend to cancel out.

In both cases, the particulars of the workload are crucial!

# Lessons Learned

- Convincing *empirical* evaluation of preference-directed caching impossible due to lack of preference data; difficult to evaluate different schemes cost-effectively.
- Workload-independent performance analysis of biased removal policies (competitive analysis) yields results too weak to guide engineering design.
- Periodic auction model that *is* IC yields poor results due to clash with event-driven dynamic of caching.
- It appears very difficult to design strategy-proof biased replacement policies.