Lecture 31

CS625: Advanced Computer Networks
Fall 2003

Friday, 07 November 2003

Bhaskaran Raman
CSE, IIT-Kanpur

http://www.cse.iitk.ac.in/users/braman/courses/cs625-fall2003/outline.html

Topic for today

- Web cache sharing
- *Scribe for today?*

Web Caching

- Purposes:
  - Reduce network bandwidth consumption
  - Reduce server load
  - Reduce client latency
- Found to be very effective, especially proxy-based caching
- Cache sharing?

Cache Sharing: ICP

- ICP: Internet Cache Protocol
  - Local cache miss ==> multicast query to all other caches
  - Improves cache hit-ratio
  - Communication and processing overhead
  - Huge overhead even for a set of 4 caches
  - How to reduce the overhead?

Alternative: Summary-Cache

- Maintain compact summary of cache directory
- On local miss, query only those caches which potentially have the web page
- Two sources of overhead
  - False-hit, false-miss
- Two issues to resolve:
  - When to do summary updates?
  - How to summarize?
- Two factors limiting scalability
  - Network overhead, memory

Impact of Update Delays

- Delay summary update until *X* % of cache documents are "new"
- *X* = 0.1 %, 1 %, 2 %, 5 %, 10 %
- Trace-driven simulations
- Delay threshold of 1-10 % works well in practice
- Translates to update frequency of about once in 5 minutes

Summary Representations

- Summary needs to be in main memory
- Memory size is a bottleneck
- Two simple possibilities:
  - Exact-Directory
    - Store 16-byte MD5 hash of URL
    - Too much memory requirement
  - Server-name
    - Store only server name
    - Too many false-hits

Bloom Filter

- Represent a set A = {a1, a2, ... an} to support membership queries
- Allocate vector of *m* bits
- Choose hash functions h1, h2, ... hk with range [1,m]
- For each element ai, mark bits h1(ai), h2(ai), ... hk(ai)
- False-positives possible
- Choose *k*, *m* such that false-positive probability is small

Bloom Filter: Choosing *k* and *m*

- Insert *n* keys ==> probability of a bit being 0 is p = (1-1/m)^kn
- Probability of false positive: (1-p)^k
  - Approximately (1-e^(kn/m))^k
  - Minimized when k is ln2 X (m/n)
  - Minimum value is 1/2^k = (0.6185)^(m/n)
- Probability decreases exponentially with m/n
  - Load factor alpha = (m/n) = # bits per data item
  - For alpha=10, k=4, false-positive prob. is 1.2%

Using Bloom Filters for Summary-Cache

- Hash on URL
- Should also support changes to set A
  - Maintain counter with each bit
  - 4 bits sufficient in practice
- Proxy builds bloom filter, sends to other proxies
- Load factor of 8 or 16 sufficient in practice
  - Same hit ratio as exact directory
- Scalability: small memory requirement even for 100 proxies