Approximate Caches for Packet Classification

Francis Chang, Kang Li, Wu-chang Feng {francis, kangli, wuchang}@cse.ogi.edu

The Motivation:

Packet Classifiers are getting more complex and Flow Identifiers are getting more unwieldy (IPv4->IPv6) So Packet Classification Caches are getting bigger and slower

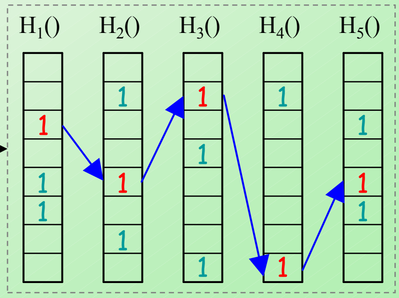
The Story:

What if we give up accuracy – let's accept some occasional mistakes. This allows us to save memory and increase performance!

Behind the scenes:

A Bloom filter optimized for packet classification can store flow identifier signatures.

Inserting a flow ID into a Bloom filter



Storing Forwarding Paths:

A Bloom filter can only store 1 bit of information – set membership. For applications more sophisticated than firewalls, we can store more information by using multiple Bloom filters.

So, by using an approximate caching strategy, we can build a cache architecture that's faster, and more memory efficient than existing exact caching strategies.

Optimizing a Bloom filter: This is the traditional Bloom filter equation – minimizing misclassification probability for a fixed # of elements.

$$p_{misclassification} = \left(1 - \left(1 - \frac{L}{M}\right)^k\right)^L$$

L=# hash levels, M=amount of memory, p= probability, k=# of elements (flows).

We prefer to maximize the # of elements for a fixed misclassification probability.

$$\kappa = -\frac{M}{L}\ln(1 - p^{1/L})$$

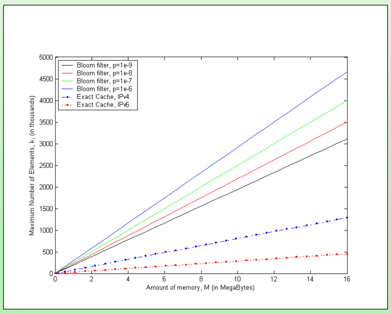
The interesting properties of a Bloom Filter:

1) The # of elements that we can fit in a Bloom filter is scales linearly with the amount of memory

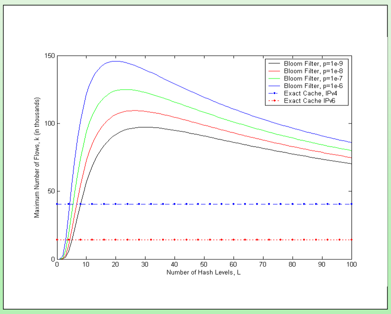
2) The optimal # of hash levels in Bloom filter is dependent only on the misclassification probability, not the amount of memory: $L = -\log_2 p$

3) For a misclassification probability of 1 in a billion, optimal dimensioning is L = 30 hash levels

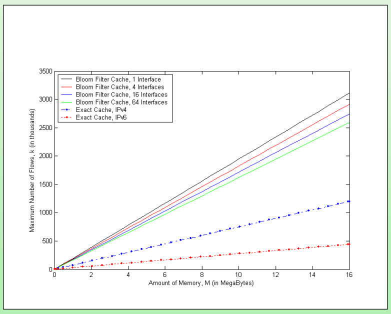
Number of flows we can fit into a Bloom filter cache: Using different misclassification probabilities, compared to an exact cache.



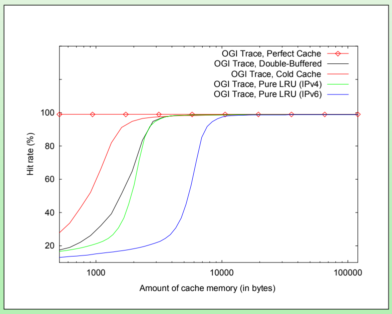
Bloom filter caches using different number of hash functions: The curve is very smooth near the optimal point.



The # of flows we can store, if we use multiple Bloom filters: The decrease in # of flows is approx. logarithmic with # of Bloom filters



The Payoff: The cache hit rate comparing Bloom filter caching and traditional exact caching.



OGI SCHOOL OF SCIENCE & ENGINEERING
OREGON HEALTH & SCIENCE UNIVERSITY

http://www.cse.ogi.edu/sysl/

Sponsored by Intel