Information-Centric Networking Research Group Meeting, Paris

A distributed **latency-aware** caching mechanism for networks of caches

The network is a big cache

Léonce Mekinda

Orange Labs Networks, Télécom Paristech

September,27 2014

Joint work with
Luca Muscariello, Orange Labs Networks
Giovanna Carofiglio , Bell Labs | Alcatel-Lucent

Caching in brief

Definition of Caching :

Any use of memory for shortening data retrieval paths

- Getting data from such memories (caches) must be faster than retrieval from original providers
- Conversely, caches are deemed much smaller than the content offer (the catalog)

⇒ Some local rationale must govern the choice of the objects worth being cached i.e. cache management policies

But none was *designed* to make any cache synchronization *emerge*

From networks of caches to cache-networks

- Network of caches $\triangleq$ self-sufficient caches that happen to be interconnected
- Cache-networks $\triangleq$ self-sufficient caches whose interconnection purposely reveal a better cache.
  As unity spontaneously makes strength.

$\Rightarrow$ We need a distributed cache management policy that :

1. *Zoom in* : greedily focuses on user QoE
2. *Zoom out* : leads to an effective object placement : avoids caching objects that are quickly served by neighbour caches.

Why probing latency in cache-networks

For both local and global purposes

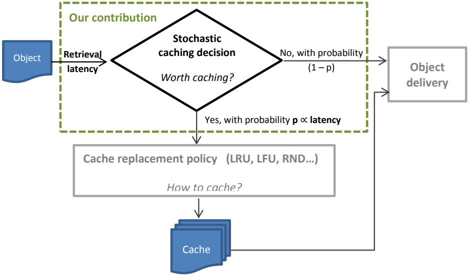
1. To minimize object retrieval cost
   - Retrieval latency $\equiv$ Round-Trip Time $\approx$ haul distance $\approx$ path load and congestion
2. It is the cheapest way to detect neighbour caches
   - Objects from neighbour caches often have insignificant latencies

$\Rightarrow$ The related objective function is

$$Minimize \sum_{i \in Catalog} Popularity(i)\mathbb{E}[RTT_i]MissProb(i)$$

$\Rightarrow$ Holistic policy : Send in priority to cache **long-to-retrieve popular** objects

Latency-aware stochastic caching decisions



Where the probability $p_k(t)$ of caching content k at time t is

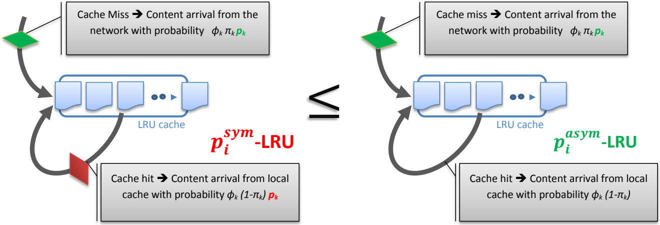
$$p_k(t) \propto \min\left(\frac{(RTT_k(t))^{\beta}}{\left(f\left((RTT_i(u))_{i\in Contents, u<t}\right)\right)^{\gamma}}, 1\right) \quad (1)$$

- β and γ are intensity parameters
- $f \equiv$ median or weighted mean or maximum

Properties (1)

Analytically tractable properties :

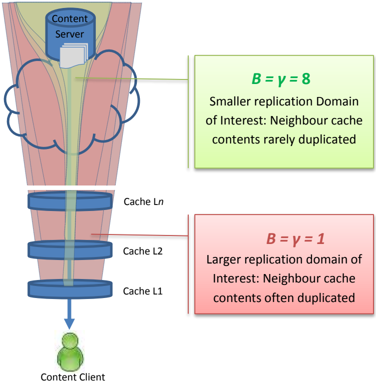
1. QoE focused $\surd$ *Eq.(1)*
2. Better hit ratio than LRU. Asymptotically mimics LFU $\surd$ As
   (i) Starobinsky-Tse-Jelenković-Radovanović 's p_i^{sym}-LRU $\sim$ LRU
   (ii) p_i^{asym}-LRU $\sim \mathbb{E}[p]^{asym}$-LRU $\underset{\mathbb{E}[p]\to 0}{\sim}$ LFU
   (iii)



in terms of number of highly popular contents ‘permanently’ stored in each cache.

Properties (2)

3. Distributed ✓
   – Effective object placement emerging from shrunk Replication DOI inside cache clusters



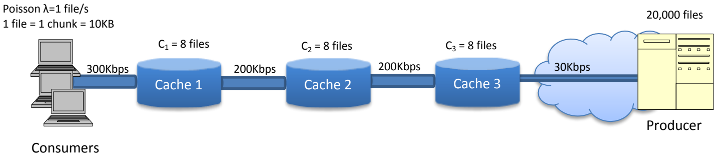
Observe it at work

Empirically observed properties :

4. Fast convergence towards a much smaller mean delivery time (smaller by up to **50%**) $\surd$
5. Stability $\surd$ Collapse of delivery time variance

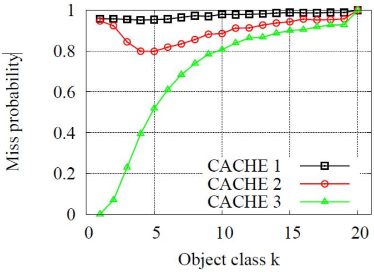
$f \equiv$ mean ; $\beta = \gamma = 8$

- **Simulation 1** : Line topology : LRU vs Latency-aware LRU



Line topology results (1)

- Miss probability



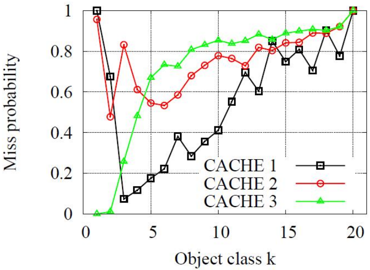


Chart 1: LRU

Chart 2: Latency-aware LRU

Line topology results (2)

- Delivery time

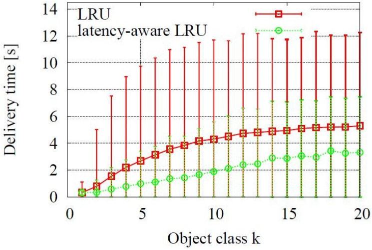


Chart 3: Delivery time with confidence interval

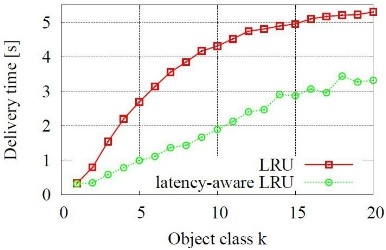


Chart 4: Mean delivery time

Line topology results (3)

- Delivery time evolution

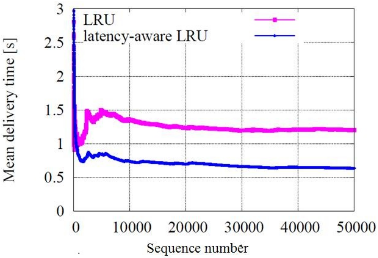


Chart 5: Mean delivery time

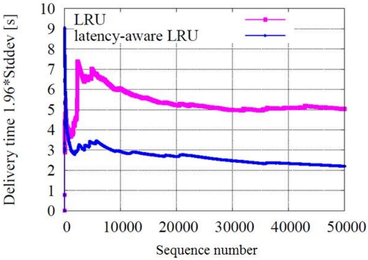
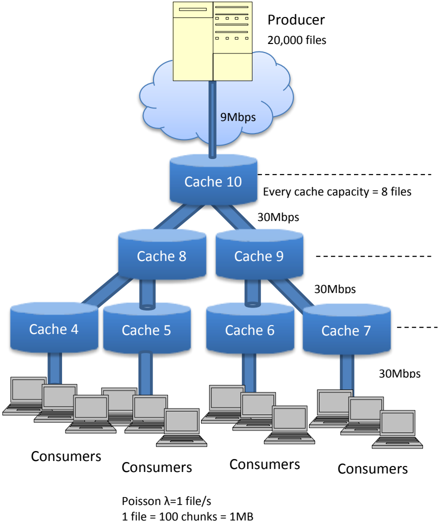


Chart 6: Delivery time StdDev

Observe it at work

- **Simulation 2** : Tree topology : LRU vs Latency-aware LRU



Tree topology results (1)

- Miss probability

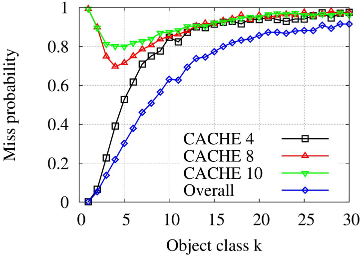


Chart 7: LRU

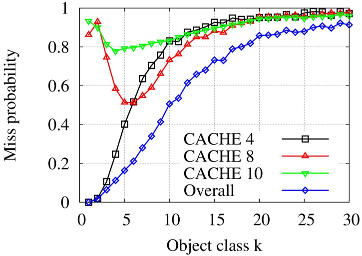


Chart 8: Latency-aware LRU

Tree topology results (2)

- Delivery time

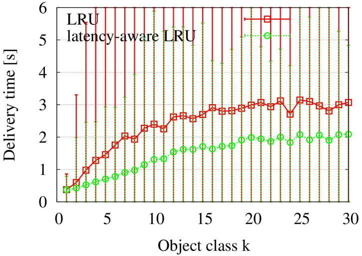


Chart 9: Delivery time with confidence interval

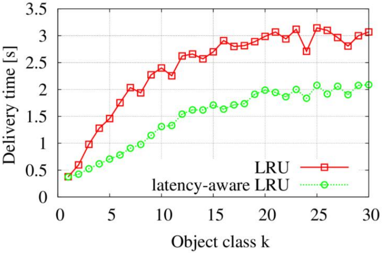
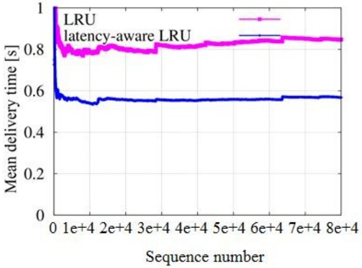


Chart 10: Mean delivery time

Tree topology results (3)

- Delivery time evolution



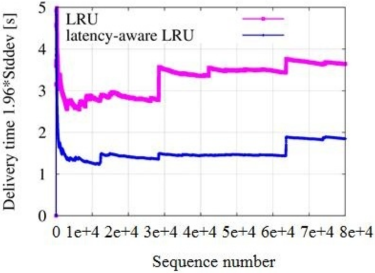


Chart 11: Mean delivery time

Chart 12: Delivery time StdDev

Tree topology results (4)

- **Simulation 3** : Same tree topology : p^{asym}-LRU vs Latency-aware LRU
  - $p = 0.05$
- Miss probability

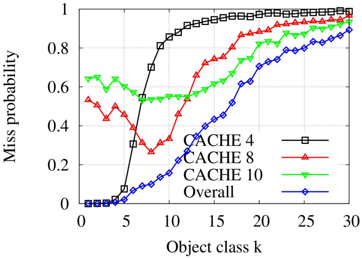


Chart 13: p^{asym}-LRU

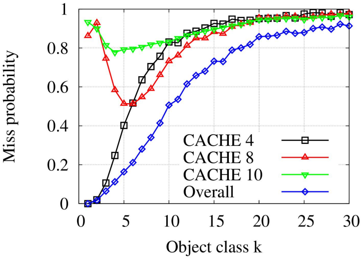


Chart 14: Latency-aware LRU

Tree topology results (5)

- Delivery time evolution

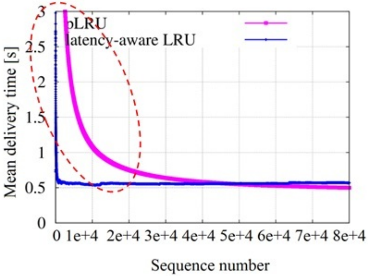


Chart 15: Mean delivery time

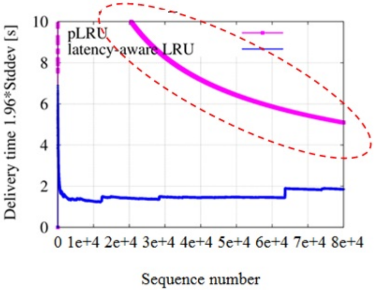


Chart 16: Delivery time StdDev

WARNING

Observe p^{asym}-LRU slow convergence vs Latency-aware LRU

Lessons learned and perspective

- The network is a big cache, cache on purpose !
- Fast convergence towards LFU is achievable by fluctuating p_i^{asym}-LRU probabilities smartly so that $p \in \{1, \epsilon(t)\}$ and $\mathbb{E}[p] \to 0$
- Future work :
  - Implement in NFD
  - Large scale deployment and test
  - Decision probability fine tuning (parameter estimation)
  - Closed-form of the hit ratio

Thank you for your attention.
Any question ?