Adaptive TTL-Based Caching for Content Delivery

Soumya Basu[1], Aditya Sundarrajan[2], Javad Ghaderi[3], Sanjay Shakkottai[1], and Ramesh Sitaraman[2,4]

UT Austin[1], UMass Amherst[2], Columbia University[3], and Akamai Technologies[4]

Content Delivery Networks (CDN)

- CDNs deliver **millions of requests** from content provider to users.
- CDNs bring **content closer to end users** by caching content locally.
- CDNs handle **heterogeneous and ephemeral** contents, e.g. webpage, video.

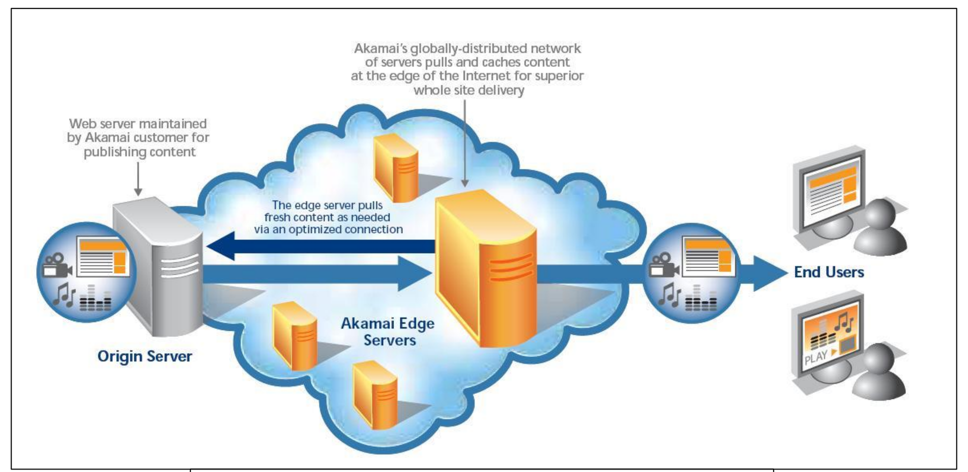

Fig 1: Content Delivery Networks

CDN Objectives and Cache Design

- **Primary Objective:** Guarantying Quality of Service (QoS) to the end users.

QoS ➡ Local Availability ➡ Cache Hit rate*

- **Ambitious Objective:** Enabling Pricing Schemes for content providers.

QoS + Dedicated Resources ➡ Local Availability + Dedicated Memory ➡ Cache Hit rate* + Cache Size*

* Explained in Caching Terminology (to the right)

Challenges in Cache Design

Content Request Process

1. Millions of Objects spanning thousands of types
2. Correlated Arrivals with complex Inter-arrival distribution
3. Complex and non-independent content popularity
4. Non-stationary arrivals – (i) One-hot wonders, (ii) Flash Crowds

- **Cache Design:** Trace-based methods and Theoretical methods

Theoretical methods

$$\textit{Design Parameter} = f\left(\textit{Hit rate, size rate, Request Process param.}\right)$$

1) Very few results with **non-stationarity**
2) **Knowledge of Model** for closed form
3) Request parameters **high-dimensional**
4) Learning is **complex** and **expensive**

Trace-based Model

Exhaustive search for parameters

Evaluation on traces via simulation

1) **Expensive** in large scale, such as CDN
2) Erroneous for **non-stationary** traces
3) Inaccurate for traces with **diverse types**
4) **Long traces** necessary for accuracy

Overcoming Challenges: Adaptive Caching

- **Model-oblivious** and **Target-driven**, **Online Adaptation** of the parameters
- **Time-to-live (TTL)** caches for adaptation with hit-rate guarantees
- **Circumventing non-convexity: Achieving size-rate , not** optimizing
- Higher degrees of freedom for size-rate control: **Two level TTL caches**.
- **Adaptive Filtering** of Non-stationary content to limit wastage of size

Dynamic TTL Cache (d-TTL)

Single Level TTL Cache

- One 'TTL', θ_t for each type t
- On **Miss**, cache with TTL, θ_t
- On **Hit**, reset the TTL to θ_t
- On timer **Expiry**, evict object

Single Level TTL Adaptation

- Hit-rate target for each type t, $h_t < 1$
- Adaptive TTL, $\theta_t(l)$ on l^{th} request
- On l^{th} request: Type t object, $\alpha \in (0.5,1)$
  - If **Miss**, $\boldsymbol{\theta_t(l) = \theta_t(l-1) + \frac{1}{l^\alpha} h_t}$
  - If **Hit**, $\boldsymbol{\theta_t(l) = \theta_t(l-1) + \frac{1}{l^\alpha}(h_t - 1)}$

Caching Terminology

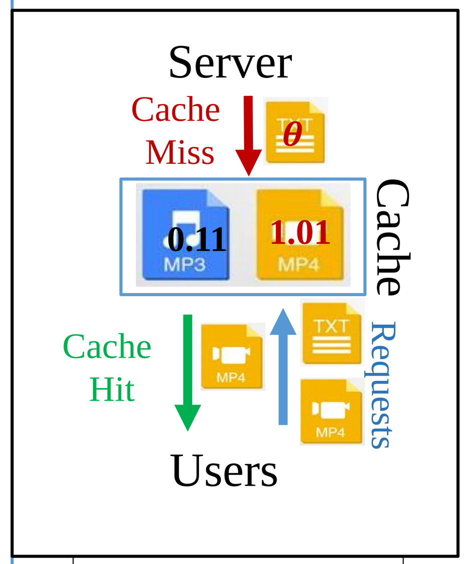

Fig 2: d-TTL Cache

Meta-data: Id of an Object.

Type: E.g. data, audio, video

Hit(i): Object in cache (i), i=1,2

Miss: Object not in (any) cache.

Virtual Hit: Object not in both cache, but object-id in cache (2).

Hit rate:$= \frac{\# \textit{Cache hit}}{\# \textit{Requests}}$

Size rate:$= \frac{\textit{Avg. cache size (Gb)}}{\textit{Arrival rate (Gbps)}}$

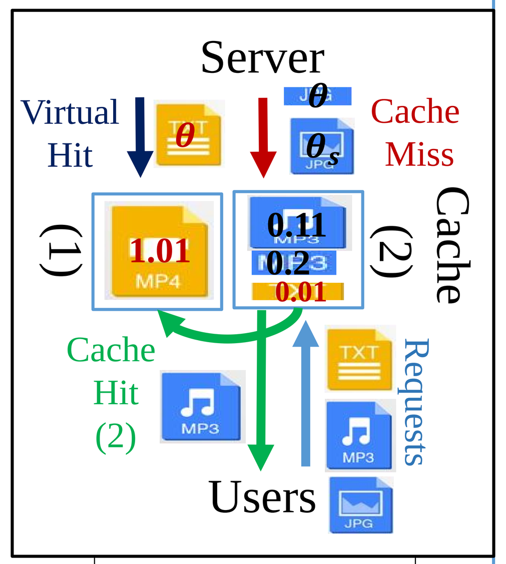

Fig 3: f-TTL Cache

Filtering TTL Cache (f-TTL)

Two Level TTL Cache

- Two level of caches: Cache (1) and (2)
- 'TTL' pair, (θ_t, θ_t^s) for each type t
- 'TTL' pair satisfies $\theta_t \geq \theta_t^s$ for all t
- On Miss,
  - i) Cache object in (2) with timer θ_t^s
  - ii) Cache meta-data in (2) with timer θ_t
- On Hit in cache (1), reset the timer to θ_t
- On Hit in cache (2),
  - i) Cache object in (1) with timer θ_t
  - ii) Evict objet and meta-data from (2)
- On Virtual hit,
  - i) Cache object in (1) with timer θ_t
  - ii) Evict meta-data from (2)
- On timer **Expiry**, evict object/meta-data

Two Level TTL Adaptation

- Hit-rate target for each type t, $h_t < 1$
- Size-rate target for each type t, s_t
- Adaptive TTL,$(\theta_t(l), \theta_t^s(l))$ on l^{th} request.
- On l^{th} request: Type t object, $\alpha \in (0.5,1)$
  - If Miss or Virtual hit,

$$\boldsymbol{\theta_t(l) = \theta_t(l-1) + \frac{1}{l^\alpha} h_t}$$

$$\boldsymbol{\theta_t^s(l) = \theta_t^s(l-1) + \frac{1}{l}(s_t - \theta_t^s(l-1))}$$

  - If Hit, denote the timer value as $\psi > 0$

$$\boldsymbol{\theta_t(l) = \theta_t(l-1) + \frac{1}{l^\alpha}(h_t - 1)}$$

$$\boldsymbol{\theta_t^s(l) = \theta_t^s(l-1) + \frac{1}{l}(s_t + \psi - \theta_t^s(l-1))}$$

  - $\boldsymbol{\theta_t^s(l) = \min\{\theta_t(l), \theta_t^s(l)\}}$

Performance on Akamai Traces

- **Akamai trace**: Duration, **9 days**, #Requests, **504m**, #Objects, **25m**.
- **Targeted hit rate** for d-TTL and f-TTL: 40%, 50%, 60%, 70% and 80%.
- **Targeted size rate** for f-TTL: 50% of size rate of d-TTL.
- **ACCURACY: Error in Achieved hit rate: 1.3%, Achieved size rate: 2%.**

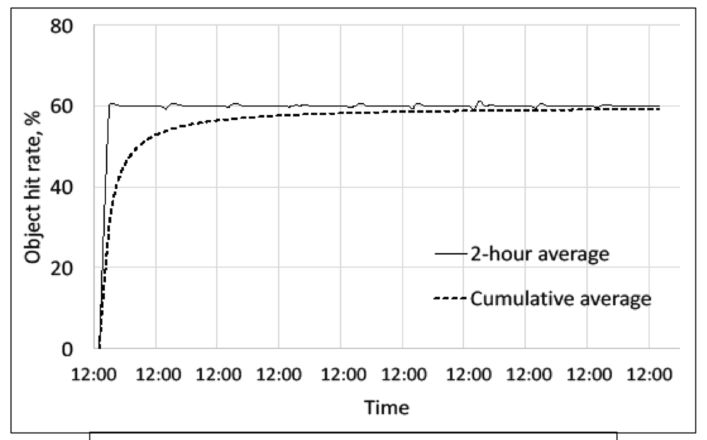

Fig 4: d-TTL Convergence Plot

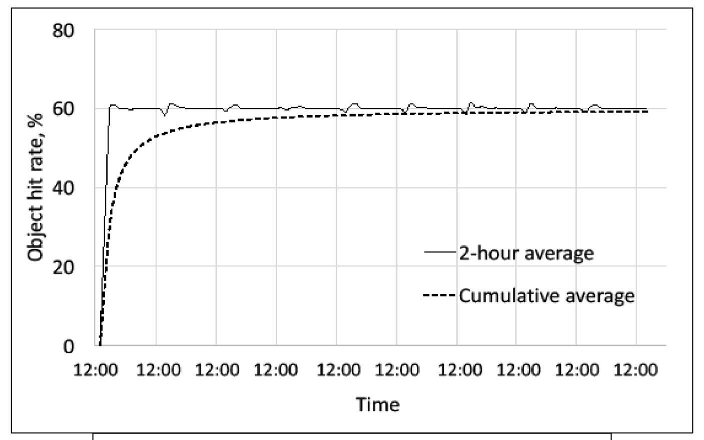

Fig 5: f-TTL Convergence Plot

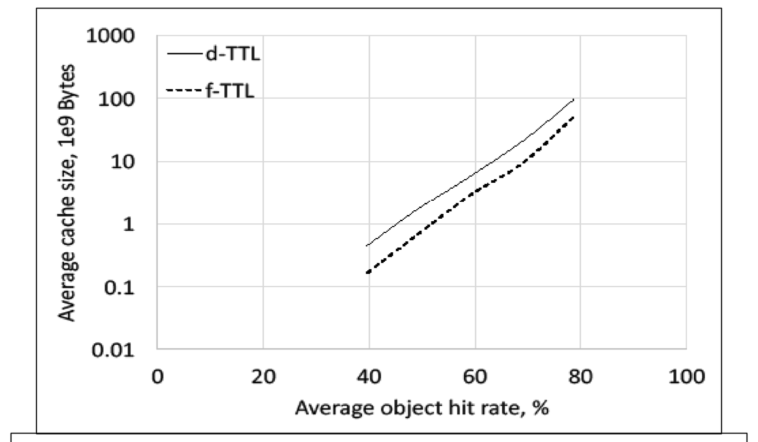

Fig 6: Hit rate vs Average Cache Size curve

Adaptation Insights

- The **larger** the value of $\boldsymbol{\theta_t}$ the **higher** the **hit-rate**.
- On **cache hit decrease** $\boldsymbol{\theta_t}$ value and on **cache miss increase.**
- The second 'TTL', $\boldsymbol{\theta_t^s}$, enables **filtering** of **rare** objects.
- Lower $\boldsymbol{\theta_t^s}$ + hit-rate target ⇒**Smaller** cache **(2)** but **larger** cache **(1)**.
- Thumb Rule: **Filtering reduces total size** under high non-stationarity.

Performance Guarantee

System Model

- Finite no. of types and finite no. of **'recurrent'** objects of each type
- Arrival of **'rare'** objects at a **non zero rate**, e.g. flash-crowd, one-hot objects
- Content request modeled as **Markovian Arrival Process**
- Inter arrival time with any **absolutely contd. pdf** with **connected support**

Performance

- d-TTL* and f-TTL* **attains the target hit-rate** asymptotically almost surely
- f-TTL* **attains size-rate ≤ the target** or **collapses** to $\boldsymbol{\theta_t^s = 0}$ a.a.s.

* A modified version of the current d-TTL and f-TTL algorithm

Proof Techniques

- **Stochastic approximation** technique used for TTL adaptation
- **Projected ODE** based proof technique to show convergence of d-TTL
- **Two timescale Actor-Critic** framework for convergence of f-TTL