Caching Policies for Delay Minimization in Small Cell Networks with Joint Transmissions

Guilherme I. Ricardo [1,2]
Giovanni Neglia [2]   Thrasyvoulos Spyropoulos [1]

[1]EURECOM, Communication Systems Department

[2]Inria, Université Côte d'Azur

Agenda

Motivation

Content Distribution Networks

- Scenario : Increasing mobile and cellular data usage.
- Question : How to provide better QoS under such scenario ?
- Solution : Content replication closer to final user - Caching !

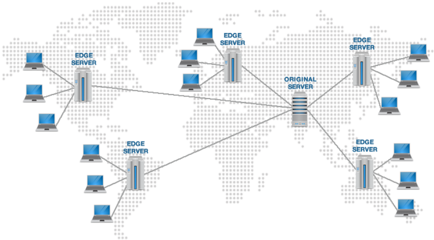


FIGURE – CDN Multiserver Caching Strategy – Source

Single Server Caching

Introduction

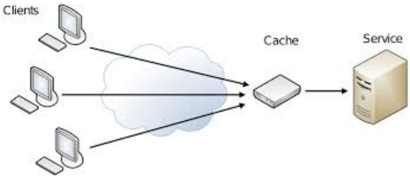


FIGURE – Single Server Caching – Source

- Problem : What to cache ?
- Performance metric : Hit Ratio
- Popularity is known : Store the most popular contents
- Popularity is unknown/dynamic : Caching algorithms (policies)

Single Server Caching

Policies Examples - Least Frequently Used (LFU)

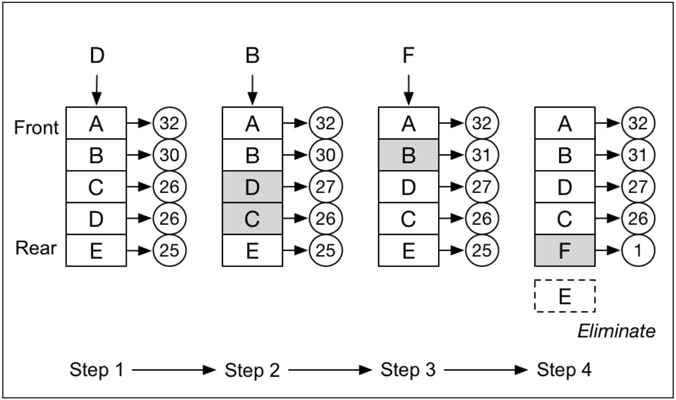


FIGURE – LFU Caching Policy – Source

Single Server Caching

Policies Examples - Least Recently Used (LRU)

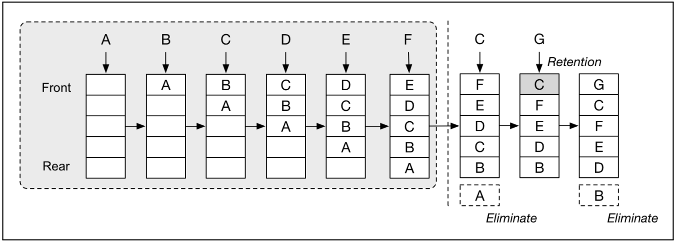


FIGURE – LRU Caching Policy – Source

- Variations :
  - qLRU – probabilistic insertion, $0 \leq q \leq 1$
  - kLRU – multilevel cache, $k = 1, 2, \ldots$

FemtoCaching Problem

5G Heterogeneous Networks Topology

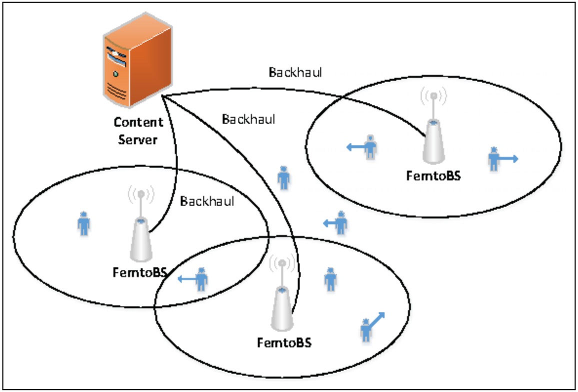


FIGURE – Heterogeneous Network Topology – Source

FemtoCaching Problem

The Optimization Formulation

Let $\mathbf{X}$ be the allocation matrix such that $x_{hf} = 1$ if helper h caches content f and $x_{hf} = 0$ otherwise. The problem is :

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & F(\mathbf{X}) = \frac{1}{U} \sum_{f=1}^{F} p_f \sum_{u=1}^{U} \mathbb{1}_{\{k(u,f)>0\}} \\ \text{subject to} \quad & \sum_{f=1}^{F} x_{hf} = C, \; h = 1, \ldots, H, \end{aligned}$$

where F is the catalog size, C is the cache capacity, U is the number of users, $k(u,f) \triangleq \sum_{h \in \mathcal{H}(u)} x_{hf}$, and $\mathcal{H}(u)$ is the set of helpers covering user u.

FemtoCaching Problem

The Offline Solution – Femto (2015)

- NP-Hard Problem (Combinatorial Nature)
- Greedy Algorithm :
  - $F(\mathbf{X})$ is monotone and submodular
  - Constraints form a matroid partition
  - 1/2-Approximation ratio
- Drawbacks : Strong assumptions, e.g.,
  - Centralized intelligence
  - Network topology and popularities are static and known

FemtoCaching Problem

The Online Solutions – Caching Policies

- LRU-One and LRU-All – Giovanidis (2016)
- qLRU-Lazy – Neglia (2018)

qLRU-Lazy Policy Description

1. Only the helper that served the file can update its cache ; and
2. It only does so if it is the only one able to actually serve it

Cooperative Multipoint Systems

Joint Transmissions and The Delay Metric

Definition

The delay $d(u, f, \mathbf{X})$ for user u to download content f under allocation $\mathbf{X}$ is

$$d(u, f, \mathbf{X}) = \begin{cases} d_B + \frac{M}{W \log_2(1+\max_h g_{hu})}, & \text{if cache miss} \\ \frac{M}{W \log_2(1+\sum_h g_{hu} x_{hf})}, & \text{if cache hit,} \end{cases}$$

where d_B is the backhaul delay, g_{hu} is the SNR from h to u, M is the file size, and W is the channel bandwidth.

Cooperative Multipoint Systems

The Optimization Problem : Formulation – Tuholukova (2018)

Delay Minimization Problem

$$\underset{x}{\text{minimize}} \quad F(\mathbf{X}) = \frac{1}{U}\sum_{f=1}^{F} p_f \sum_{u=1}^{U} d(u, f, \mathbf{X})$$

$$\text{subject to} \quad \sum_{f=1}^{F} x_{hf} = C,\ h = 1, \ldots, H$$

Remark

Submodularity Proof and Greedy Algorithm

Cooperative Multipoint Systems

Hit Ratio → Avg. Delay

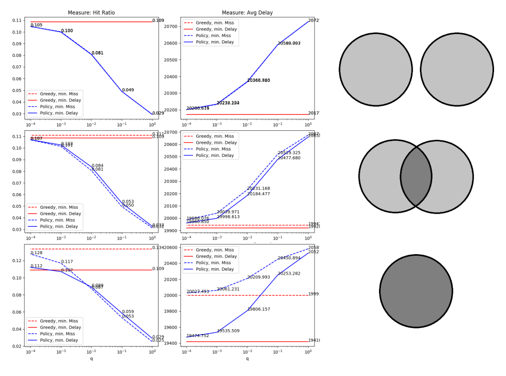


FIGURE – Static allocation for different overlapping levels (full rep.)

Cooperative Multipoint Systems

Optimal Allocation : d_B x SNR Bounds

Assumptions :

- Completely overlap
- Homogeneous SNR (γ)

For a given γ, if $d_B \geq d_{B,max}$ such that

$$d_{B,\max}(C,H,\alpha,\gamma) \triangleq (HC)^{\alpha}\frac{M}{W}\left(\frac{1}{\log_2(1+\gamma)} - \frac{1}{\log_2(1+2\gamma)}\right)$$

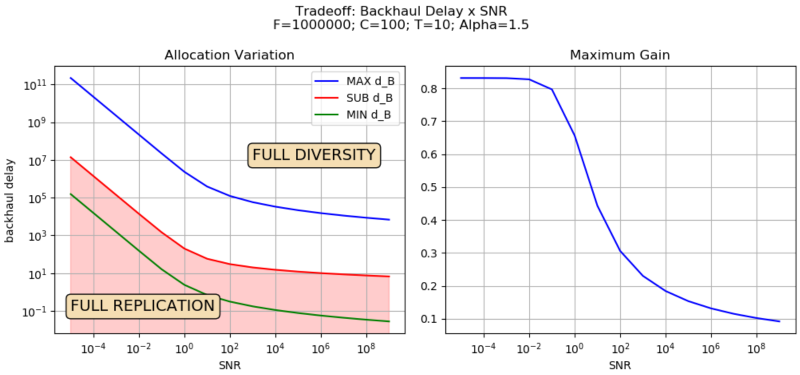
then the optimal allocation is full diversity.
For a given γ, if $d_B \leq d_{B,min}$ such that

$$d_{B,\min}(C,H,\alpha,\gamma) \triangleq \left(\frac{C+1}{C}\right)^{\alpha}\frac{M}{W}\left(\frac{1}{\log_2(1+(H-1)\gamma)} - \frac{1}{\log_2(1+H\gamma)}\right)$$

then the optimal allocation is full replication.

Cooperative Multipoint Systems

Optimal Allocation : d_B x SNR Bounds, Example



CoMP Caching Algorithms

qLRU-Δd Policy Notation

- Let I_u be the set of helpers covering user u and $J_{u,f} \subseteq I_u$ be the subset of those helpers caching f.
- The marginal gain for adding a copy of file f at helper h is defined as :

$$\Delta d^{(h)}(u, f, \mathbf{X}) \triangleq d(u, f, \mathbf{X} \ominus \mathbf{e}^{(h)}) - d(u, f, \mathbf{X})$$

- Normalizers :

$$\beta \triangleq 1/(\max_{f,h,u,\mathbf{X}} \Delta d^{(h)}(u, f, \mathbf{X}))$$

$$\gamma \triangleq 1/(\max_{f,h,u,\mathbf{X}} \Delta d^{(h)}(u, f, \mathbf{X} \oplus \mathbf{e}^{(h)})).$$

CoMP Caching Algorithms

qLRU-Δd Policy Introduction

qLRU-Δd Policy General Description

At every request (u, f), each $h \in I_u$ updates its cache as follows :

- If $h \in J_{u,f}$, reset f's cache position with probability :

$$\rho^{(h)}(u, f, \mathbf{X}) = \beta \cdot \Delta d^{(h)}(u, f, \mathbf{X})$$

- If $h \in I_u \backslash J_{u,f}$, store f to h's cache with probability $q \cdot \sigma^{(h)}(u, f, \mathbf{X})$, where $q \in (0, 1]$ is fixed and

$$\sigma^{(h)}(u, f, \mathbf{X}) = \gamma \cdot \Delta d^{(h)}(u, f, \mathbf{X} \oplus \mathbf{e}^{(h)})$$

CoMP Caching Algorithms

qLRU-Δd Policy Introduction

qLRU-Δd Policy Algorithmic Description

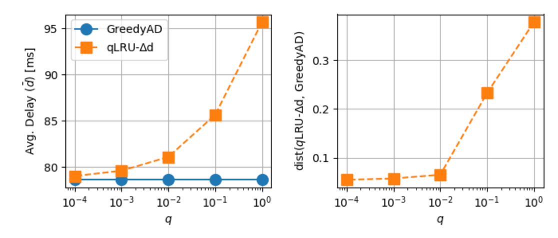
Input: I_u, $J_{u,f}$, and $g_{h',u}, \forall h' \in I_u$
for $h \in I_u$ **do**
  if $h \in J_{u,f}$ **then**
    Move f to the front with prob. $\rho^{(h)}$
  else
    Evict file in the cache's last position;
    Insert f with prob. $q \cdot \sigma^{(h)}(u, \mathbf{X}_f)$.
  end
end

CoMP Caching Algorithms

qLRU-Δd Policy Introduction

Remark – Ricardo (2020)

Under IRM, Che's, and Exponentialization approximations, a network of qLRU-Δd caches converges to a locally-optimal caching configuration when $q \to 0$.



CoMP Caching Algorithms

2LRU-Δd Policy Notation

- IRM $\neq$ Real request process (Temporal locality)
- Each helper deploys a 2-levels cache : the physical cache storing the actual file and the virtual cache storing files' metadata (i.e., ID)
- Let I_u be the set of helpers covering user u and let $J_{u,f}, \hat{J}_{u,f} \subseteq I_u$ be the subsets of those helpers storing f at the physical cache and at the virtual cache, respectively.

CoMP Caching Algorithms

2LRU-Δd Policy Introduction

2LRU-Δd Policy General Description

At every request (u, f), each $h \in I_u$ updates its cache as follows :

- If $h \in \hat{J}_{u,f}$, move f's ID to the front of h's virtual cache and,
  - if $h \in J_{u,f}$, move f to the front of h's physical cache with prob. $\rho^{(h)}(u, f, \mathbf{X})$;
  - else, evict the file in the physical cache's last position and insert f.
- If $h \notin \hat{J}_{u,f}$, with prob. $q \cdot \sigma^{(h)}(u, f, \mathbf{X})$, evict the ID in h's virtual cache's last position and insert f's ID

CoMP Caching Algorithms

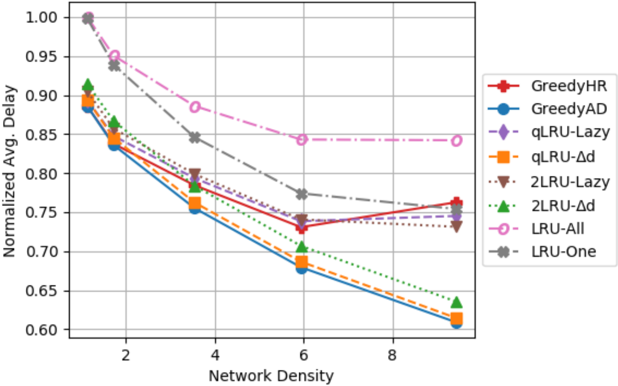
2LRU-Δd Policy Introduction

2LRU-Δd Policy Algorithmic Description

Input: I_u, $J_{u,f}$, $\hat{J}_{u,f}$, and $g_{h',u}, \forall h' \in I_u$
for $h \in I_u$ **do**
  if $h \in \hat{J}_{u,f}$ **then**
    Move f's ID to the front of the virtual cache;
    if $h \in J_{u,f}$ **then**
      Move f to the front of the physical cache with prob. $\rho^{(h)}$
    else
      Evict file in physical cache's last position;
      Insert f.
    end
  else
    Evict file's ID in virtual cache's last position;
    Insert f's ID with prob. $\sigma^{(h)}$.
  end
end

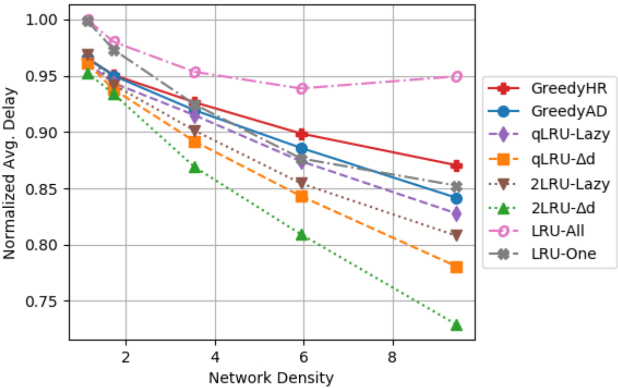
CoMP Caching Algorithms

Numerical Results – IRM, Homogeneous SNR



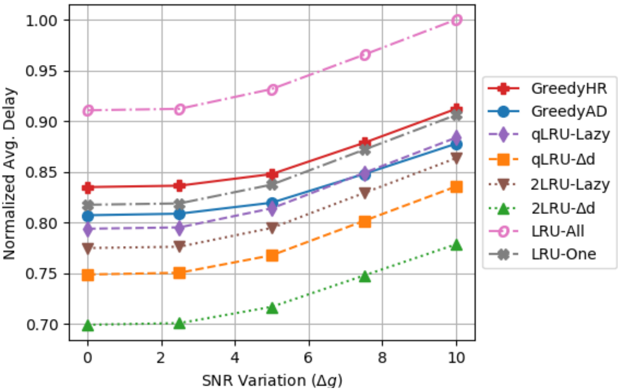
CoMP Caching Algorithms

Numerical Results – Real, Homogeneous SNR



CoMP Caching Algorithms

Numerical Results – Real, Heterogeneous SNR



Conclusion and Future Works

- Conclusions
  - Delay cost function under CoMP provides different allocation with potentially better download rates
  - qLRU-Δd Policy outperforms other Hit Ratio dynamic policies for synthetic requests
- Future Work
  - Finish Real Traces Experiments
  - Greedy Algorithm with pair of files
  - Finish Algorithm

Thank You!