Fair throughput allocation in Information-Centric Networks

Thomas Bonald, Léonce Mekinda, Luca Muscariello

Fair throughput allocation in Information-Centric Networks

Thomas Bonald[a], Léonce Mekinda[b], Luca Muscariello[c]

[a]*TELECOM ParisTech, 23 Avenue d'Italie, 75013 Paris, France*
[b]*European XFEL, Holzkoppel 4, 22869 Schenefeld, Germany*
[c]*Cisco Systems, 11 Rue Camille Desmoulins, 92130 Issy-les-Moulineaux, France*

Abstract

Cache networks are the cornerstones of today's Internet, helping it to scale by an extensive use of Content Delivery Networks (CDN). Benefiting from CDN's successful insights, ubiquitous caching through Information-Centric Networks (ICN) is increasingly regarded as a premier future Internet architecture contestant. However, the use of in-network caches seems to cause an issue in the fairness of resource sharing among contents. Indeed, in legacy communication networks, link buffers were the principal resources to be shared. Under max-min flow-wise fair bandwidth sharing [14], content throughput was not tied to content popularity. Including caches in this ecosystem raises new issues since common cache management policies such as probabilistic Least Recently Used (p-LRU) or even more, Least Frequently Used (LFU), may seem detrimental to low popularity objects, even though they significantly decrease the overall link load [3]. In this paper, we demonstrate that globally achieving *LFU is a first stage of content-wise fairness*. Indeed, any investigated content-wise α-fair throughput allocation permanently stores the most popular contents in network caches by ensuring them a cache hit ratio of 1. As ICN caching traditionally pursues LFU objectives, content-wise fairness specifics remain only a matter of fair bandwidth sharing, keeping the cache management intact.

Keywords: ICN, Caching, Fairness, Network Performance Analysis.

1. Introduction

Today's Internet owes its scalability to caching. Indeed, most of Internet contents cross Content Delivery Networks and significant research is pushing for a better solution, Information-Centric Networks. In ICN, and more specifically, Named-Data Networking (NDN) and Content-Centric Networking (CCN) [9], two leading ICN architectures, content objects are identified by their unique name. At every node/router, content Data packets are requested via matching Interest packets, through egress interfaces. Interests and their satisfying Data counterparts follow rigorously the same path. This feature would not be possible without the Pending Interest Table (PIT) structure

Email addresses: thomas.bonald@telecom-paristech.fr (Thomas Bonald), leonce.mekinda@xfel.eu (Léonce Mekinda), lumuscar@cisco.com (Luca Muscariello)

that keeps track of every requesting interface and requested content. Naming Data packets allows storing them, on every traversed node, in a finite memory referred to as Content Store (CS) or cache and managed by an object eviction policy.

Caches and their eviction or management policies are the disruption that drives this paper. Traditionally, networks are modeled as interconnected queues with fair schedulers. The penetration of caching into the network layer clearly favors a few content objects, the most popular ones in case of the Least Frequently Used management policy (LFU) and its approximations such as (p-)LRU or LRU+Leave-Copy-Down [13]. Filling caches steadily with the most popular items, meaning keeping their hit ratio to their maximum *i.e.,* one, and letting other hit ratios be zero, entails the sacrifice of less popular objects [3]. This is at least a view discussed by state-of-art contributions on content-wise cache fairness [25] [6]. These works observed the hit ratio on a single cache or a network of caches and prescribed an adaptation of the cache management policy for the purpose of fairness. For example, in [6], content-wise max-min fairness is only achievable if the hit ratios are forced to be equal for all content objects. In the same vein, proportional fairness requires that content hit ratio be proportional to their popularity. A consequence of this is that ICN cannot be fair to contents without revising its caching algorithms. From the viewpoint of these works, LFU is definitely unfair to lower popularity contents. By the way, remember *flow-wise* fairness means allocating resources such that every flow/route gets its fair share. On the other hand, by *content-wise* fairness, we denote allocating resources in such a way every content gets its fair share. This is the type of fairness this paper addresses.

Our paper analyzes the fairness of content delivery throughput in accounting for both cache hit ratio and link service rates, and comes up with a different and optimistic conclusion. *ICN's traditional caching optimum leads to content-wise fairness as it is*. The better the convergence to LFU, the better the feasible content-wise fairness. The remaining task would consist in implementing content-wise fairness at the packet scheduling stage in ICN, similarly to flow-wise fairness in other networks [10]. Taking a network of caches as a whole, links and caches, the paper sheds new light on content-wise fair cache allocation. While previous works only considered caches and concluded that caching policies have to be adapted to be α-fair to contents, this work shows that LFU and its approximations are sufficient as they are, and content-wise α-fairness is the responsibility of network packet schedulers. This contribution brings α-fairness in ICN and α-fairness in traditional networks closer. Our results owe to the link service to the majority of contents that balances the rather permanent cache presence of a few contents. It is rather commonplace that persisting the most popular contents frees a maximal upstream link capacity to convey less popular objects. Another striking insight we got, is that a throughput-optimal content delivery network ends up being made up of autonomous caches that never forward their miss traffic. Such a network would not be committed to locally satisfy requests.

The main contributions of this paper are that: (i) it unifies caches and network queues into a single content service rate model; (ii) it tackles for the first time content throughput fairness in ICN in formulating that as a tractable nonlinear optimization problem; (iii) it provides closed-form expressions of α-fair hit ratios and link service rates; (iv) it indicates that today's LFU-approximating caches policies do not need to be replaced for ICN to become fair. We articulate these contributions throughout the

paper as follows: Sec.2 recapitulates previous contributions on fairness in the context of cache networks. In Sec.3, we model the per-content throughput in unifying cache and network link contributions. Then we formalize α-fair allocations, key properties such as their Pareto-efficiency, and that LFU is an α-fair cache management policy, an important result. To ground the theory, a few trivial examples are analyzed in Sec.4. They are followed in Sec.5 with numerical evaluations that confirmed, by means of a nonlinear problem solver, our analytic insights.

2. Related work

Very few papers address the issue of fairness in networks of caches. In a paper dedicated to the subject some time ago [25], authors analyze the fairness in Content-Centric Networks from the viewpoint of object dissemination across the network. They expressed content-wise fairness as the total space contents occupy with respect to their popularity. The study concluded that medium-popularity content were favored as they spread linearly with their popularity whereas the most popular items spread sub-linearly. This approach is definitely useful to map the asymptotic replica spatial distribution. However, it does not capture the throughput fairness. [21] and [22] tackled the impact of fairness on delivery time in large scale CDN but ignored the cache specifics. That work modeled cache networks as classical networks of file-serving queues. Files were assumed to have been pre-fetched and their long-term popularity was not taken into account.

Quite recently, [6] reverse-engineered popular LRU and LFU policy and found the utility function each policy optimizes. These utility functions achieve various classes of hit ratio α-fairness. Authors also provided algorithms for adapting Time-to-Live (TTL)-based caches to any given α-fair objective. Rapidly, [16] applied this work's reverse engineering approach to a special case of a novel class of latency-aware caching (LAC) policies previously introduced by [5]. In [16], authors show that LAC policies converge to the solution of a fractional knapsack problem (LFU) when their latency exponent tends to infinity.

Most of the existing literature on the subject, because of its focus on hit ratio, concluded that caching policies had to adapt to the content-wise fair objective. Our contribution is novel because it joins cache and link queue occupation in order to analyze the QoE-expressive throughput fairness. The QoE considered in the paper refers to how fair the user may perceive the throughput of the most popular content compared to those of less popular contents. We show that cache networks, and ICN in particular, can be α-fair, for any $\alpha \geq 0$, as soon as they couple the classical highest popularity content persistence *i.e.,* the global LFU cache management policy, with a proper content-aware α-fair packet scheduler.

3. Cache Network Model

First, we present a mathematical model that captures the dynamics of the entire network. The model views the latter as a network of queues where caches contribute to increase the network service rate. We aim at maximizing a utility function of the

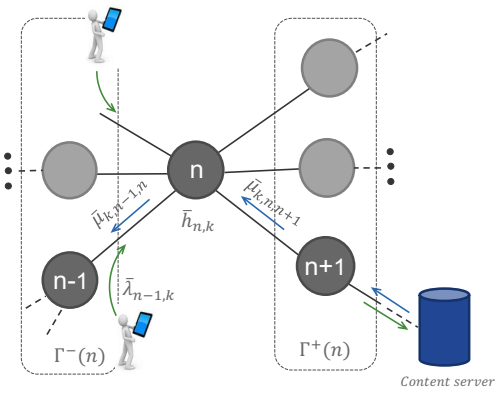

Figure 1: Network conveying content k through cache n.

admissible exogenous traffic rate. Refer to Table 1 for the notation and to Fig.1 for the model used hereinafter.

3.1. Model assumptions

- Let the stochastic process $\{\lambda_{k,n,b}(t)\}_{0\leq t\leq T}$ be content k exogenous rate on link (n,b) at time t. Let the stochastic process $\{\mu_{k,n,b}(t)\}_{0\leq t\leq T}$ be content k service rate on the link (b,n) at time t. Let the stochastic process $\{h_{n,k}(t)\}_{0\leq t\leq T}$ be content k hit ratio on node n at time t. These processes are independent.
- The network routes based on a single prefix.
- Same object sizes. This is a widely adopted assumption in the caching literature [7]. It lies on the idea that the actual disparities among content sizes are embodied by the popularity factor q_k, which multiplies a content quantum (*e.g.,* a mean chunk size)
- Cache size is never zero.
- Content servers are not clients.
- The exogenous traffic on a given node is the one generated by a local application that is not satisfied by the local cache.
- We assume hop-by-hop congestion control *i.e.,* interests are sent in average at a rate equivalent to the link service rate.

Let us define a Pending Interest Queue (PIQ) size as the number of pending interests per content and per interface. An interest queued in a PIQ is served when the matching data packet comes back.

$n \in \mathcal{N}$	ICN node identifier. $\mathcal{N} \subseteq \mathbb{N}$.
$t \in \mathbb{R}_+$	Instant a content retrieval occurs.
$k \in \mathcal{K}$	Content popularity rank. The one ranking first is the most popular, while rank $\|\mathcal{K}\|$ indicates the least popular object.
$\Gamma^-(n)$	Set of node n's ingress nodes.
$\Gamma^+(n)$	Set of node n's egress nodes.
$\bar{\mu}_{k,n,b}$	Long-term average of the service rate for content k on link (b, n)
λ_n	Long-term average of exogenous interest rate at node n.
$\lambda_{n,k}$	Long-term interest rate for content k at node n.
q_k	Content k popularity. It is the probability that a requested content is content k. It is strictly ordered: $q_{k+1} < q_k$.
$\mathbb{1}_{\{\cdot\}}$	Indicator function.
$\Lambda_{n,k}$	Upper bound for the long-term average of exogenous interest rate for content k at node n .
$\varsigma(k)$	Set of content k servers.
$\bar{h}_{n,k}$	hit ratio of content k at node n
$C_{b,n}$	Link (b, n) capacity in chunks/s.
x_n	Cache n capacity in objects.

Table 1: Notation.

Let $Q_{k,n,b}(t)$ be the size of the Pending Interest Queue for content k at time t for link (n, b). $h_{n,k}(t) \equiv \mathbb{1}_{\{k \text{ in cache } n \text{ at } t\}}$ indicates whether content k was found in cache n at time t. The time evolution upper bound of the PIQ size of content k for egress nodes $b \in \Gamma^+(n)$ at node n follows:

$Q_{k,n,b}(0) = 0, \forall b \in \Gamma^+(n)$ and

$$\sum_{b\in\Gamma^+(n)} \frac{d}{dt} Q_{k,n,b}(t) \leq \lambda_{n,k}(t) + (1 - h_{n,k}(t)) \sum_{a\in\Gamma^-(n)} \mathbb{1}_{\{Q_{k,a,n}(t)>0\}} \mu_{k,a,n}(t) - \sum_{b\in\Gamma^+(n)} \mathbb{1}_{\{Q_{k,n,b}(t)>0\}} \mu_{k,n,b}(t). \quad (1)$$

The service rate $\mu_{k,a,b}(t)$ of the PIQ is the data rate for content k on link (b, a) at time t. $\lambda_{n,k}(t) \equiv q_k \lambda_n(t)$ is the exogenous interest rate for content k at node n at time t. After some algebra, the maximum admissible rate $\bar{\Lambda}_{n,k}$ is given by:

$$\bar{\Lambda}_{n,k} \equiv \sum_{b\in\Gamma^+(n)} \bar{\mu}_{k,n,b} - (1 - \bar{h}_{n,k}) \sum_{a\in\Gamma^-(n)} \bar{\mu}_{k,a,n}, \forall n, k \quad (2)$$

constrained by the following bounds:

$$\sum_{k\in\mathcal{K}} \bar{\mu}_{k,a,n} \leq C_{n,a}, \quad \forall n, a \in \Gamma^-(n) \quad (3)$$

$$\sum_{k\in\mathcal{K}} \bar{h}_{n,k} = x_n, \quad \forall n \quad (4)$$

$$0 \leq \bar{h}_{n,k} \leq 1, \quad \forall n, k \quad (5)$$

$$\bar{\mu}_{k,a,n} \geq 0, \quad \forall n, k, a \in \Gamma^-(n) \quad (6)$$

$$\bar{\lambda}_{n,k} \leq \bar{\Lambda}_{n,k}, \quad \forall n, k. \quad (7)$$

We ignore throughout the paper constraint (7) that imposes a lower bound to the content service rate. It means that the network will not guarantee that some content requested on a given node will be satisfied. This will entirely depend on the optimality of serving that content.

3.2. Cache network capacity

The network provides a content delivery service through the coupling of disseminated caches and the links interconnecting them. The following equation unifies in a single expression the maximum service rate the network can deliver given cache hit ratios and link capacities.

It arises by first summing all maximum admissible rates at node n:

$$\begin{aligned}\sum_k \bar{\Lambda}_{n,k} &= \sum_k \Big[\sum_{b\in\Gamma^+(n)} \bar{\mu}_{k,n,b} - (1-\bar{h}_{n,k}) \sum_{a\in\Gamma^-(n)} \bar{\mu}_{k,a,n} \Big] \\ &= \sum_{k,b\in\Gamma^+(n)} \bar{\mu}_{k,n,b} - \sum_{k,a\in\Gamma^-(n)} (1-\bar{h}_{n,k}) \bar{\mu}_{k,a,n}.\end{aligned}$$

Define $\bar{\boldsymbol{\mu}}_n^-$ as the ingress rate matrix $(\bar{\mu}_{k,a,n})_{k,a}$ and $\bar{\boldsymbol{h}}_n$ as the column vector $(\bar{h}_{n,k})_k$. It follows that

$$\begin{aligned}\sum_{k,a\in\Gamma^-(n)} \bar{\mu}_{k,a,n}\bar{h}_{n,k} = \|{}^t\bar{\boldsymbol{\mu}}_n^-\bar{\boldsymbol{h}}_n\|_1 &\le \|{}^t\bar{\boldsymbol{\mu}}_n^-\|_1 \|\bar{\boldsymbol{h}}_n\|_1 \\ &\le \sup_k \Big\{ \sum_{a\in\Gamma^-(n)} \bar{\mu}_{k,a,n} \Big\} x_n,\end{aligned}$$

where ${}^t\bar{\boldsymbol{\mu}}_n^-$ is the transpose of the ingress rate matrix and $\|{}^t\bar{\boldsymbol{\mu}}_n^-\|_1$ is the operator norm [2] associated to the Banach space ℓ^1 *i.e.,* $(\mathbb{R}^{|\mathcal{K}|}, \|\ \|_1)$, applied to the ingress rate matrix.

Then we sum all maximum admissible rates. Rates that are both egressing from a node and ingressing to another node vanish. We obtain:

$$\begin{aligned}\sum_{n,k} \bar{\Lambda}_{n,k} &\le \sum_{\substack{k,b\in\varsigma(k)\\ n\in\Gamma^-(b)}} \bar{\mu}_{k,n,b} + \sum_n \sup_k \Big\{ \sum_{a\in\Gamma^-(n)} \bar{\mu}_{k,a,n} \Big\} x_n \\ &\le \sum_{\substack{b\in\cup_k\varsigma(k)\\ n\in\Gamma^-(b)}} C_{b,n} + \sum_n \sup_k \Big\{ \sum_{a\in\Gamma^-(n)} \bar{\mu}_{k,a,n} \Big\} x_n. \qquad (8)\end{aligned}$$

3.3. Problem formulation

We now plug the admissible rate into a fair utility function $U(\cdot)$. Define the network wide allocated rate for content k as

$$\phi_k \equiv \sum_{n\in\mathcal{N}} \bar{\Lambda}_{n,k} = \sum_{\substack{b\in\cup_k\varsigma(k)\\ n\in\Gamma^-(b)}} \bar{\mu}_{k,b,n} + \sum_{\substack{n\in\mathcal{N}\\ a\in\Gamma^-(n)}} \bar{h}_{n,k}\bar{\mu}_{k,a,n}. \qquad (9)$$

The problem's objective is to find the optimal link service rates and hit ratios that

$$\underset{\bar{\boldsymbol{\mu}},\bar{\boldsymbol{h}}}{\text{maximize}} \sum_{k\in\mathcal{K}} q_k U(\phi_k/q_k), \tag{10}$$

given the α-fair utility function $U(\cdot)$. Weighted α-fairness was first introduced in [15], and later adapted for cache allocation by [6]. However, as expressed in Eq.10, we advocate for a formulation of *weighted α-fairness* that operates on rates per weight unit. The reason for this is its convergence to *weighted max-min* fairness as $\alpha \to \infty$, whereas the original Mo and Walrand's weighted formulation decays into max-min fairness [15]. Interestingly, as shown later in the paper, our formulation gives solutions that are independent of α, shaping as such, just fair allocations.

$$\text{Since } q_k U(\phi_k/q_k) \equiv q_k \frac{(\phi_k/q_k)^{1-\alpha}}{1-\alpha} = q_k^{\alpha}\frac{\phi_k^{1-\alpha}}{1-\alpha}, \alpha \neq 1,$$

the objective simplifies to

$$\underset{\bar{\boldsymbol{\mu}},\bar{\boldsymbol{h}}}{\text{Maximize}} \quad \begin{cases} \sum\limits_{k\in\mathcal{K}} q_k^{\alpha} U(\phi_k), & \text{if } \alpha \neq 1 \\ \sum\limits_{k\in\mathcal{K}} q_k \log(\phi_k/q_k), & \text{otherwise.} \end{cases} \tag{11}$$

Special cases this weighted α-fairness framework encompasses are:

- for $\alpha = 1$, the objective is then said to be weighted-proportional fair [10].
- for an infinite value of α, the objective is weighted max-min fair *i.e.,* $\max\min(\phi_k/q_k)$ [17].

In the rest of the document, the attribute *weighted* will be implied when ommitted. Define the vectors of decision variables $\bar{\boldsymbol{\mu}} \equiv (\bar{\mu}_{k,b,n})$ and $\bar{\boldsymbol{h}} \equiv (\bar{h}_{n,k})$. Also, define the vector of multipliers $\boldsymbol{\nu} \equiv \left(\nu_{\cdot}^{(i)} \geq 0\right)$ where (i) identifies the constraint. The Lagrangian of the problem is

$$\begin{aligned} \mathcal{L}(\bar{\boldsymbol{\mu}},\bar{\boldsymbol{h}},\boldsymbol{\nu}) = &\sum_k q_k^{\alpha} U(\phi_k) - \sum_{n,a\in\Gamma^-(n)} \nu_{n,a}^{(1)}\Big[\sum_k \bar{\mu}_{k,a,n} - C_{n,a}\Big] \\ &- \sum_{n,k} \nu_n^{(2)}\Big[\sum_k \bar{h}_{n,k} - x_n\Big] - \sum_{n,k} \nu_{n,k}^{(3)} \bar{h}_{n,k}(\bar{h}_{n,k}-1) \\ &+ \sum_{n,a,k} \nu_{k,a,n}^{(4)} \bar{\mu}_{k,a,n}. \end{aligned} \tag{12}$$

Although $U(\cdot)$ is non-decreasing and concave, as ϕ_k is non-concave, the Karush-Kuhn-Tucker (KKT) conditions are simply necessary for optimality. The first-order KKT conditions command that

$$\nabla_{\bar{\boldsymbol{\mu}},\bar{\boldsymbol{h}}} \mathcal{L}(\bar{\boldsymbol{\mu}}^*,\bar{\boldsymbol{h}}^*,\boldsymbol{\nu}^*) = \vec{0}, \tag{13}$$

where $\nabla_{\bar{\boldsymbol{\mu}},\bar{\boldsymbol{h}}}\mathcal{L}$ is the gradient of function $\mathcal{L}$ with respect to vectors $\bar{\boldsymbol{\mu}}$ and $\bar{\boldsymbol{h}}$. $\bar{\boldsymbol{\mu}}^* \equiv (\bar{\mu}^*_{k,b,n})$, $\bar{\boldsymbol{h}}^* \equiv (\bar{h}^*_{n,k})$, $\boldsymbol{\nu}^* \equiv (\nu^{(i)*}_{\cdot})$ are the optimal counterparts of the aforementioned vectors.

3.4. Solution

3.4.1. General α-fair allocation

For any $\alpha \geq 0$, the Lagrangian expands as follows:

$$\begin{aligned}\mathcal{L}(\bar{\boldsymbol{\mu}},\bar{\boldsymbol{h}},\boldsymbol{\nu}) = \frac{1}{1-\alpha}\sum_k q_k^\alpha \Big[\sum_{\substack{b\in\cup_k \varsigma(k)\\ n\in\Gamma^-(b)}} \bar{\mu}_{k,b,n} + \sum_{\substack{n\in\mathcal{N}\\ a\in\Gamma^-(n)}} \bar{h}_{n,k}\bar{\mu}_{k,a,n}\Big]^{1-\alpha}\\ - \sum_{n,a\in\Gamma^-(n)} \nu^{(1)}_{n,a}\Big[\sum_k \bar{\mu}_{k,a,n} - C_{n,a}\Big] - \sum_{n,k}\nu^{(2)}_n\Big[\sum_k \bar{h}_{n,k} - x_n\Big]\\ - \sum_{n,k}\nu^{(3)}_{n,k}\bar{h}_{n,k}(\bar{h}_{n,k}-1) + \sum_{n,a,k}\nu^{(4)}_{k,a,n}\bar{\mu}_{k,a,n}.\end{aligned}$$

The first property of content-wise α-fair allocations in cache networks is their Pareto efficiency. An allocation is said to be Pareto efficient if any attempt to increase one content's share decreases the share of another content. In our optimization problem, this translates into link capacity being fully allocated.

Property 1 (Pareto efficiency for any $\alpha \geq 0$)**.** The α-fair bandwidth allocation is Pareto efficient as the optimal resource uses the whole link capacities to serve content items *i.e.,*

$$\sum_{k\in\mathcal{K}} \bar{\mu}^*_{k,a,n} = C_{n,a}, \quad \forall n \in \mathcal{N}, \forall a \in \Gamma^-(n). \tag{14}$$

Proof. See in Appendix Appendix A. □

Then comes our main result. It established that ICN, in adopting the Least Frequency Used as the caching policy maximizing content hit ratio, has de facto adopted an optimal caching for content throughput fairness.

Proposition 1 (LFU leads to α-fairness)**.** LFU is a cache management policy for a network seeking α-fairness, for any $\alpha \geq 0$.

Proof. See in Appendix Appendix B. □

This result is important as it shows that the LFU algorithm and its heuristics (LRU, p-LRU, LRU-k, LRU-LCD) can lead to α-fair networks, $\forall\alpha \geq 0$. Packet schedulers would be in charge of the other part of the optimal solution: bandwidth sharing that is fair to contents. We refer to the latter as content-wise α-fair bandwidth sharing. It is mathematically tractable thanks to the concavity of the problem, given binary hit ratios, as concavity is a sufficient condition for the existence of a global optimum. Furthermore, content-wise α-fair bandwidth sharing is practically achievable within the ICN paradigm as packets are uniquely named after the content they carry.

We may also emphasize the novelty of this result, since previous works [6] reached very different conclusions. This is because they only looked at isolated caches, found that fairness required fractional hit ratio for each value of the fairness parameter α greater than zero, and suitably designed algorithms for TTL-based caches. Their caching algorithms consist in adjusting every content Time-To-Live (TTL) via gradient descent.

To summarize, the following algorithm (Alg.1, Alg.2) is an example of distributed content-wise weighted max-min fairness implementation. It relies on a Deficit Round-Robin scheduler [24] to achieve content-wise max-min fair bandwidth allocation, given the LFU caching substrate.

Algorithm 1: Content-wise α-fair allocation in ICN

Input: Data packet, α
Cache.Insert(packet, Policy::LFU);
FairQueuing.Shape(packet, α);

Algorithm 2: Content-wise weighted max-min fair bandwidth sharing

function `FairRate.Shape` (*Data packet,* α)
  FairQueuing.Queue[packet.ContentName()].Push(packet);
  if $\alpha == \infty$ **then**
    FairQueuing.SendData(Policy::DEFICIT_ROUND_ROBIN);
  end
end

4. Toy examples

We analyze two trivial cases to foster some further insight on the preceding results, and preclude limit case quandaries. We tackle the case of a connected client/server tandem network then we illustrate the fairness problem on a client/cache/server bus.

4.1. Client/Server tandem network

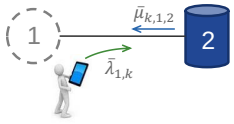

Figure 2: Client/server topology.

A communication link conveys some exogenous traffic from a client node numbered 1 to a content server numbered 2. There is no cache in between. The α-fair objective is:

$$\text{Maximize} \quad \sum_k q_k^\alpha \frac{(\bar{\mu}_{k,1,2})^{1-\alpha}}{1-\alpha}.$$

The optimal allocation for every content k, for any value of $\alpha \geq 0$, is

$$\mu_{k,1,2} = q_k C_{2,1}.$$

4.2. Client/Cache/Server bus

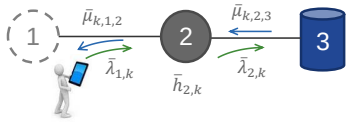

Figure 3: Client/Cache/Server bus topology.

In this toy scenario, exogenous traffic at a client node 1 is conveyed towards a content server 3 through an intermediate cache 2.

4.2.1. Proportional fairness

The related objective is

Maximize

$$\sum_k q_k \log \frac{\bar{h}_{2,k}\bar{\mu}_{k,1,2} + \bar{\mu}_{k,2,3}}{q_k}.$$

Assume that $\bar{\mu}_{k,2,3} = 0.$ *, i.e.,* the server's egress link capacity is zero. Then, the following two concave terms have to be independently maximized:

$$\sum_k q_k \log \frac{\bar{h}_{2,k}}{q_k} + \sum_k q_k \log \frac{\bar{\mu}_{k,1,2}}{q_k}.$$

The optimal solutions are $\bar{h}_{2,k} = q_k x_2$ and $\bar{\mu}_{1,k} = q_k C_{2,1}$.

If $\bar{\mu}_{k,2,3} > 0.$ we are in a situation where the server can deliver data through its ingress link. The following demonstration shows LFU is the cache heuristics that always finds the unique optimum if one exists. By the way, this claim is later confirmed numerically in Sec.5.1.

• Assuming that the optimal $\bar{h}_{2,k} \in \{0, 1\}$, we can deduce the optimal link services rates.

To that aim, first define the two sets $\mathcal{K}_i \equiv \{k \in \mathcal{K} : \bar{h}_{2,k} = i\}$, $\forall i \in \{0, 1\}$. $\mathcal{K}_0$ is the set of objects that are not stored in the cache, $\mathcal{K}_1$ is the set of object that are permanently cached. As such, $|\mathcal{K}_1| = x_2$. So, the concave objective yields

Maximize

$$\sum_{k\in\mathcal{K}_0} q_k \log \frac{\bar{\mu}_{k,2,3}}{q_k} + \sum_{k\in\mathcal{K}_1} q_k \log \frac{\bar{\mu}_{k,1,2} + \bar{\mu}_{k,2,3}}{q_k}. \tag{15}$$

Define β as an optimal multiplier tied to the constraint upon the capacity of the link to the server. If the KKT conditions hold,

$$\bar{\mu}_{k,2,3} = \frac{q_k}{\beta}, \forall k \in \mathcal{K}_0, \text{ and}$$
$$\bar{\mu}_{k,2,3} = \frac{q_k}{\beta} - \bar{\mu}_{k,1,2}, \forall k \in \mathcal{K}_1.$$

As

$$\sum_k \bar{\mu}_{k,2,3} = C_{3,2} = \frac{1}{\beta}\Big[\sum_{k\in\mathcal{K}_0} q_k + \sum_{k\in\mathcal{K}_1} q_k\Big] - \sum_{k\in\mathcal{K}_1} \bar{\mu}_{k,1,2},$$

we obtain

$$\frac{1}{\beta} = C_{3,2} + \sum_{k\in\mathcal{K}_1} \bar{\mu}_{k,1,2} \leq C_{3,2} + C_{2,1}.$$

Hence, the optimal rates satisfy

$$\bar{\mu}_{k,2,3} = q_k\Big[C_{3,2} + \sum_{k\in\mathcal{K}_1} \bar{\mu}_{k,1,2}\Big], \forall k \in \mathcal{K}_0, \tag{16}$$
$$\bar{\mu}_{k,1,2} + \bar{\mu}_{k,2,3} = q_k\Big[C_{3,2} + \sum_{k\in\mathcal{K}_1} \bar{\mu}_{k,1,2}\Big], \forall k \in \mathcal{K}_1. \tag{17}$$

• We insert that solution into Eq.15 to outline the sets $\mathcal{K}_i$. At the optimum, the objective function reaches its supremum

$$S \equiv \sup_{\mathcal{K}_1}\Big\{\log(C_{3,2} + \sum_{k\in\mathcal{K}_1} \bar{\mu}_{k,1,2}) : |\mathcal{K}_1| = x_2 \text{ and } \sum_{k\in\mathcal{K}_1} \bar{\mu}_{k,1,2} \leq C_{2,1}\Big\}. \tag{18}$$

Define the network capacity $\kappa \equiv C_{2,1} + C_{3,2}$. The upper bound of S, denoted as $S^{\max}$ equals $\log \kappa$. As illustrated in Fig.4, a greedy algorithm finds the supremum $S \leq S_{\max}$ by piggybacking the x_2-most popular objects in $\mathcal{K}_1$. LFU is this greedy heuristics. As such, it greedily chooses the most popular contents as those worth being stored into the cache. Optimally, that also implies sharing the entire cache's ingress link capacity among these contents. The following observations can be made:

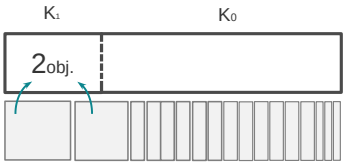

Figure 4: Example of greedy resolution of Eq.18 with $x_2 = 2$. Piggybacking into $\mathcal{K}_1$ the two most significant contents, as LFU does, is optimal.

Observation 1 (Maximal solution existence)**.** The greedy algorithm finds a cache allocation and fair service rates such that the objective function equals its upper bound $S^{\max}$ *iif* $\sum_{k=1}^{x_2} q_k \geq C_{2,1}/\kappa$.

This happens when the cumulative fair service rate for the objects in cache exceeds the capacity of the link to that cache. Even if that solution might not be unique, (for example, in case of $C_{2,1}/\kappa$ being too small), whenever $S^{\max}$ is reachable, the greedy algorithm finds a solution achieving it.

Observation 2 (Maximal solution uniqueness)**.** There exists a unique combination of cache allocation and fair service rates such that the objective function equals its upper bound $S^{\max}$ *iif* $\sum_{k=1}^{x_2-1} q_k + q_{x_2+1} < C_{2,1}/\kappa$.

Indeed, if one can not replace the least popular object stored in the cache by some other object and get a fair service rate exceeding the capacity of the cache's ingress link, then the LFU-provided cache configuration is the unique optimum.

Conversely, if the cumulative fair service rate of the x_2-most popular objects remains lower than the capacity of the cache's ingress link: $\sum_{k\in\mathcal{K}_1} \mu_{k,1,2} < C_{2,1}$. This happens when the cache's ingress link $(2,1)$ is over-provisioned. Consequently, the link service rates of never-cached objects $k \in \mathcal{K}_0$, $\mu_{k,1,2} > 0$, and the fair objective function can not reach its upper bound $S^{\max}$ as the network carries some miss traffic. By miss traffic, we denote content retrievals that are triggered by cache miss events.

• To conclude, remember $\bar{h}_{2,k}$ was assumed to equal either 0 or 1. We show that $S^{\max}$ is also the upper bound of the general objective function we denote f, for any $\bar{h}_{2,k} \in [0,1]$. Indeed, as the objective function increases with any of the decision variables, any attempt to increase $S^{\max}$ to $S^{\max}+\epsilon$, $\forall \epsilon > 0$, necessarily increases some zero hit ratio by $\delta\bar{h} > 0$ and decreases a hit ratio of one by the same amount.

$$\epsilon = \Big[\frac{\partial f(\bar{h})}{\partial \bar{h}}\Big|_{\bar{h}_{2,k}=0} - \frac{\partial f(\bar{h})}{\partial \bar{h}}\Big|_{\bar{h}_{2,k}=1}\Big]\delta\bar{h} \qquad = -\frac{\bar{\mu}_{k,1,2}}{C_{2,1}+C_{3,2}}\delta\bar{h}.$$

As $\epsilon \leq 0$, the solution provided through LFU caching is the optimum for any $\bar{h}_{2,k} \in [0,1]$.

4.2.2. General α-fairness

Here the objective consists in the following:

$$\text{Maximize} \quad \sum_k \frac{q_k^\alpha}{1-\alpha}(\bar{h}_{2,k}\bar{\mu}_{k,1,2} + \bar{\mu}_{k,2,3})^{1-\alpha}.$$

As demonstrated previously, the optimal $\bar{h}_{2,k}$ belong to $\{0,1\}$. It allows to reuse the aforementioned definition of the sets $\mathcal{K}_i$. We can calculate the optimal link services rates, owing to the concave objective function

$$\sum_{k\in\mathcal{K}_0} \frac{q_k^\alpha}{1-\alpha}(\bar{\mu}_{k,2,3})^{1-\alpha} + \sum_{k\in\mathcal{K}_1} \frac{q_k^\alpha}{1-\alpha}(\bar{\mu}_{k,1,2} + \bar{\mu}_{k,2,3})^{1-\alpha}. \tag{19}$$

The KKT conditions yield

$$\bar{\mu}_{k,2,3} = \frac{q_k}{\beta^{1/\alpha}}, \forall k \in \mathcal{K}_0, \text{ and}$$
$$\bar{\mu}_{k,2,3} = \frac{q_k}{\beta^{1/\alpha}} - \bar{\mu}_{k,1,2}, \forall k \in \mathcal{K}_1.$$

It follows that

$$\sum_k \bar{\mu}_{k,2,3} = C_{3,2} = \frac{1}{\beta^{1/\alpha}} \Big[\sum_{k \in \mathcal{K}_0} q_k + \sum_{k \in \mathcal{K}_1} q_k \Big] - \sum_{k \in \mathcal{K}_1} \bar{\mu}_{k,1,2}$$

gives the following optimal rates:

$$\bar{\mu}_{k,2,3} = q_k \Big[C_{2,1} + \sum_{k \in \mathcal{K}_1} \bar{\mu}_{k,1,2} \Big], \forall k \in \mathcal{K}_0$$
$$\bar{\mu}_{k,1,2} + \bar{\mu}_{k,2,3} = q_k \Big[C_{2,1} + \sum_{k \in \mathcal{K}_1} \bar{\mu}_{k,1,2} \Big], \forall k \in \mathcal{K}_1,$$

making the optimal allocation identical to the proportional fairness case we presented earlier.

5. Evaluation

We numerically solved problem (10) using SCIP 3.2.1 [1], a Mixed Integer Non-Linear Program (MINLP) optimization suite. It actually performs *branch-cut-and-price* on mixed integer problems and invokes the Interior Point Optimizer IPOPT 3.12.5 [26] to solve relaxed nonlinear instances. IPOPT itself relies on PARDISO 5.0.0 [12][18][19] for tackling large-scale linear systems of equations when needed. We did not used SCIP's Mixed Integer Programming features since all our decision variables are real. It has essentially been used as an interpreter to the ZIMPL mathematical language [11] and a programming interface for IPOPT.

5.1. Client/Cache/Server bus

Consider the same bus topology as in Sec.4.2 above. Consider that exogenous requests address a catalog size of 80 objects. The content popularity follows a Zipf distribution of parameter 1. The cache budget is 10 objects. The link capacity from the cache to the client is 10 objects/s while the one from the server to the cache has a 20 objects/s capacity. Fig.5 depicts the results. LFU is clearly the proportionally fair caching policy. Also, observe that link capacities are shared proportionally to content popularity. For instance, as anticipated by Eq.17, the sum of the ingress rates for content 1 is $2.9 + 3.1 = 6 = q_1 \times (C_{2,1} + C_{3,2}) = 0.2 \times (10 + 20)$.

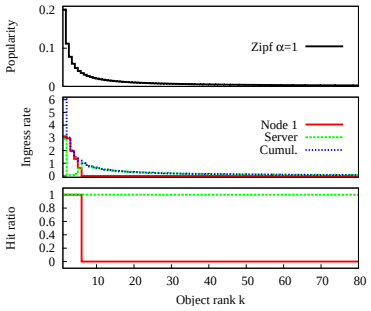

Figure 5: Bus topology: proportionally fair allocation.

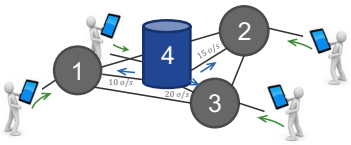

Figure 6: Simple network topology.

5.2. A simple network

Next, we evaluate a network of 3 cache-equipped routers/nodes and one server (Fig.6). The computational complexity of this nonconcave problem prevents the investigation of bigger instances. However, this setup suffices for characterizing the optima.

We set the total routers ingress link rates to respectively 10,15 and 20 objects/s. The capacity of the link to the content server is 30 objects/s. Cache capacities at every content router is 5, 6 and 7 objects. The content server is viewed as a node with a cache capacity that equals the catalog size.

The key insight is that every tractable instance entails an optimal caching consisting in the long-term storage of number of the most popular objects. This indicates that an LFU caching policy leads to content-wise α-fairness. Another implication is that at the optimum, the network does not convey any miss traffic. In other words, optimally, no interest crosses the nearest cache. Hence, the optimal network is a set of autonomous clusters centered on caches surrounded by their clients. Remember that such an optimum does not aim to guarantee interest satisfaction. We detail our observations below.

5.2.1. ($\alpha = 0$)-fairness

In the very particular case of zero-fairness, we consider a catalog of 2000 Zipf-ranked objects. Note that only the first 80 objects are depicted as the trends are clear. Beyond rank 80, objects are still neither cached nor delivered.The trivial optimum in Fig.7 depicts the whole network capacity being allocated to a single content, the most popular one. However, multiple optima exist, including a bandwidth allocation proportional to content popularity, as the problem turns out to be unweighted.

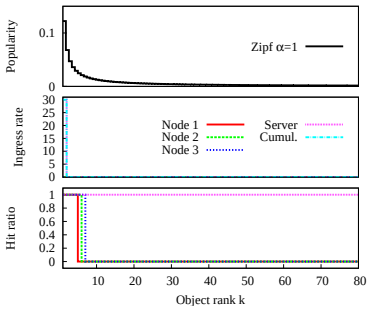

Figure 7: Simple network: A zero-fair allocation for every content k at every cache n.

5.2.2. Proportional fairness

Proportional fairness first translates into ensuring a hit ratio of 1 for a majority of the most popular content. It is distinctive in Fig.8 where the Zipf skewness is 1 and Fig.9 featuring a Zipf parameter that equals 0.7. Given that persistent caching, fairness is actually enforced by an adequate link capacity sharing. The throughput fair share follows a curve that matches the content popularity, indicating proportionality.

This is a key result as it decouples caching and scheduling in the pursuit of content-wise fairness. Cache network fairness simplifies into legacy-but-content-wise queuing network fairness, given a few additional content servers formally known as caches.

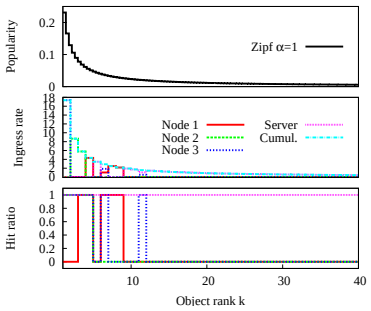

Figure 8: Simple network: proportionally fair allocation for every content k at every cache n.

5.2.3. ($\alpha = 2$)-fairness

Even when $\alpha = 2$, fairness remains consistent with the previous observations. There is no fractional hit ratio. As a consequence, content-wise 2-fairness in a cache network does not require shared-time cache occupancy. The optimal link capacity share in this case is proportional to q_k where q_k is the probability that a requested object is of popularity rank k. It shows once more that the packet scheduler is entirely in charge of content-wise fairness.

5.2.4. Max-min fairness

We evaluate numerically max-min fairness as α-fairness with $\alpha = 9$. Computational limitations prevented us from exceeding this value. Although this is a quite

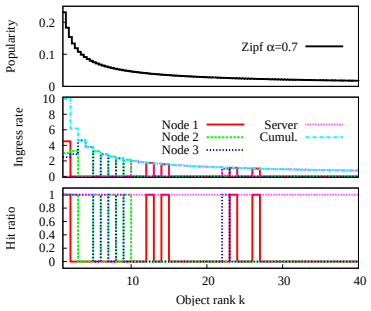

Figure 9: Simple network: proportionally fair allocation with less skewed popularity distribution.

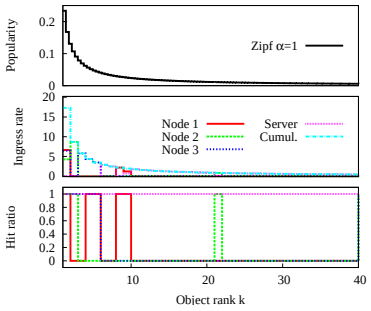

Figure 10: Simple network: ($\alpha = 2$)-fair allocation for every content k at every cache n.

loose approximation of an infinite α, the insight we get remains relevant. As before, the recipe to fairness turns out to be persistent caching and a content-wise bandwidth fair sharing on top of the classical client-server network infrastructure. Observe that, as predicted analytically, fair resource sharing remains insensitive to α. The max-min fair share, just like in proportional fair sharing or any other case, are proportional to content popularity.

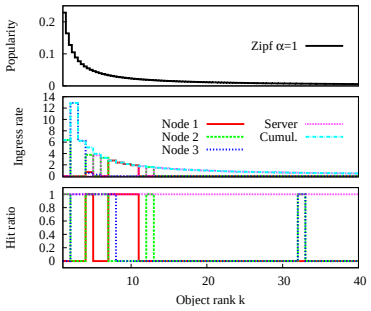

Figure 11: Simple network: Approaching max-min fairness with ($\alpha = 9$)-fair allocation for every content k at every cache n.

6. Conclusion

Cache networks, and more specifically Information-Centric Networks (ICN), are prominent solutions for communication infrastructure offloading. Early in the Internet history, cache engines have been inserted between the content consumers and the delivery servers to confine the recurrent traffic of very popular objects close the network edge. Nowadays, ICN, FemtoCaching[8] and Fog-RAN [20][23] have furthered the ongoing caching penetration.

Throughout this paper, we show that a resource allocation α-fair to content items, at any value of $\alpha \geq 0$, can solely tackle the design of fair packet schedulers while ensuring that the most popular objects get permanently cached. In contrast to previous works, that focused on isolated caches, it appears that no fractional content hit ratios is necessary for the sake of fairness.

As a strong consequence, our analytic contribution suggests that content-wise fair allocation in cache networks can be formulated within the existing frameworks pertaining to queuing networks [14] by viewing caches equipped with LFU-approximating caching policies like p-LRU, LRU+Leave-Copy-Down heuristics, or LAC+ as regular popular content servers [4]. To sum up, ICN can be α-fair to contents, as long as the link service rate allocation is α-fair.

[1] T. Achterberg. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[2] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[3] G. Carofiglio, M. Gallo, and L. Muscariello. Bandwidth and Storage Sharing Performance in Information Centric Networking. *Elsevier Science, Computer Networks Journal, Vol.57, Issue 17*, 2013.

[4] G. Carofiglio, L. Mekinda, and L. Muscariello. Analysis of Latency-Aware Caching Strategies in Information-Centric Networking. In *Proc. of ACM CoNEXT, CCDWN Workshop*, 2015.

[5] G. Carofiglio, L. Mekinda, and L. Muscariello. LAC: Introducing latency-aware caching in information-centric networks. In *Proc. of IEEE LCN*, Oct. 2015.

[6] M. Dehghan, L. Massoulie, D. Towsley, D. Menasche, and Y. Tay. A utility optimization approach to network cache design. *arXiv preprint arXiv:1601.06838*, 2016.

[7] C. Fricker, P. Robert, and J. Roberts. A versatile and accurate approximation for cache performance. In *24th International Teletraffic Congress*, 2012.

[8] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire. Femtocaching: Wireless video content delivery through distributed caching helpers. In *INFOCOM, 2012 Proceedings IEEE*, pages 1107–1115. IEEE, 2012.

[9] V. Jacobson, D. Smetters, J. Thornton, and al. Networking named content. In *Proc. of ACM CoNEXT*, 2009.

[10] F. P. Kelly, A. K. Maulloo, and D. K. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research society*, 49(3):237–252, 1998.

[11] T. Koch. *Rapid Mathematical Prototyping*. PhD thesis, Technische Universität Berlin, 2004.

[12] A. Kuzmin, M. Luisier, and O. Schenk. Fast methods for computing selected elements of the greens function in massively parallel nanoelectronic device simulations. In F. Wolf, B. Mohr, and D. Mey, editors, *Euro-Par 2013 Parallel Processing*, volume 8097 of *Lecture Notes in Computer Science*, pages 533–544. Springer Berlin Heidelberg, 2013.

[13] V. Martina, M. Garetto, and E. Leonardi. A unified approach to the performance analysis of caching systems. *CoRR*, abs/1307.6702, 2013.

[14] L. Massoulié and J. Roberts. Bandwidth sharing: objectives and algorithms. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1395–1403. IEEE, 1999.

[15] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on Networking (ToN)*, 8(5):556–567, 2000.

[16] G. Neglia, D. Carra, M. Feng, V. Janardhan, P. Michiardi, and D. Tsigkari. *Access-time aware cache algorithms*. PhD thesis, Inria Sophia Antipolis, 2016.

[17] B. Radunović and J.-Y. L. Boudec. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Transactions on Networking (TON)*, 15(5):1073–1083, 2007.

[18] O. Schenk, M. Bollhöfer, and R. A. Römer. On large-scale diagonalization techniques for the anderson model of localization. *SIAM Rev.*, 50(1):91–112, Feb 2008.

[19] O. Schenk, A. Wächter, and M. Hagemann. Matching-based preprocessing algorithms to the solution of saddle-point problems in large-scale nonconvex interior-point optimization. *Computational Optimization and Applications*, 36(2-3):321–341, 2007.

[20] A. Sengupta, R. Tandon, and O. Simeone. Cloud RAN and edge caching: Fundamental performance trade-offs,. In *Proc. IEEE International workshop on Signal Processing advances in Wireless Communications (SPAWC)*, 2016.

[21] V. Shah and G. de Veciana. Performance evaluation and asymptotics for content delivery networks. In *INFOCOM, 2014 Proceedings IEEE*, pages 2607–2615. IEEE, 2014.

[22] V. Shah and G. de Veciana. Impact of fairness and heterogeneity on delays in large-scale content delivery networks. In *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 375–387. ACM, 2015.

[23] Y. Shi, J. Zhang, K. B. Letaief, B. Bai, and W. Chen. Large-scale convex optimization for ultra-dense cloud-ran. *IEEE Wireless Communications*, 22(3):84–91, 2015.

[24] M. Shreedhar and G. Varghese. Efficient fair queueing using deficit round robin. In *ACM SIGCOMM Computer Communication Review*, volume 25, pages 231–242. ACM, 1995.

[25] M. Tortelli, I. Cianci, L. Grieco, G. Boggia, and P. Camarda. A fairness analysis of content centric networks. In *Network of the Future (NOF), 2011 International Conference on the*, pages 117–121. IEEE, 2011.

[26] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

Appendix A. Proof of α-fair allocation's Pareto efficiency

The partial derivatives with respect to the ingress capacities $\bar{\mu}_{k,a,n}$ give:

$$\bar{h}^*_{n,k} = \frac{\nu^{(1)*}_{n,a} - \nu^{(4)*}_{k,n,a}}{(q_k/\phi^*_k)^\alpha}, \forall k, n, a \in \Gamma^-(n).$$

As the sum of local hit ratios equals the size of the cache,

$$\Big[x_n + \sum_k \nu^{(4)*}_{k,a,n} \Big(\frac{\phi^*_k}{q_k}\Big)^\alpha\Big] \Big[\sum_k \Big(\frac{\phi^*_k}{q_k}\Big)^\alpha\Big]^{-1} = \nu^{(1)*}_{n,a} > 0, \forall n, a \in \Gamma^-(n).$$

As $\nu^{(1)*}_{n,a}$ is strictly positive since the cache size is strictly positive too. By the complementary slackness conditions of the convex optimization framework, the related constraint must be saturated. That translates into Eq.14.

Moreover, first derivatives with respect to the server's ingress capacities $\mu_{k,n,b}$ give:

$$\Big(\frac{q_k}{\phi^*_k}\Big)^\alpha + \nu^{(4)*}_{k,n,b} = \nu^{(1)*}_{n,b} > 0, \forall k, b \in \varsigma(k), n \in \Gamma^-(b).$$

The multipliers being strictly positive, the corresponding constraints must be saturated. It makes Eq.14 hold and Pareto efficiency follow. □

Appendix B. Proof that LFU leads to α-fairness

Let $f(\cdot)$ be the α-fair objective function. The increase of f with regards to an increase of content k's ingress rate is

$$df_{\mu_{k,a,n}} = \frac{\partial f(\bar{\mu}^*_{k,a,n})}{\partial \bar{\mu}^*_{k,a,n}} d\bar{\mu} = \bar{h}^*_{n,k} \Big(\frac{q_k}{\phi^*_k}\Big)^{\alpha} d\bar{\mu}.$$

Let $\epsilon(\alpha)$ be the increase of the α-fair objective function induced by an increase of content 1's rate and the equivalent decrease of content k's rate, $k \uparrow \infty$.

$$\epsilon(\alpha) \equiv df_{\mu_{1,a,n}} - \lim_{k\to\infty} df_{\mu_{k,a,n}} = \Big[\bar{h}^*_{n,1} - \lim_{k\to\infty} \bar{h}^*_{n,k} \Big(\frac{q_k/\phi^*_k}{q_1/\phi^*_1}\Big)^{\alpha}\Big] d\bar{\mu}. \tag{B.1}$$

We aim at proving that $\nu^{(3)*}_{n,k} > 0, \forall n, k$. Due to KKT complementary slackness conditions, it would imply that $\bar{h}^*_{n,k} \in \{0, 1\}, \forall n, k$. The partial derivatives w.r.t. cache hit ratios give

$$\sum_{a\in\Gamma^-(n)} \bar{\mu}^*_{k,a,n} = \Big(\frac{\phi^*_k}{q_k}\Big)^{\alpha} \Big[\nu^{(2)*}_n + \nu^{(3)*}_{n,k}(2\bar{h}^*_{n,k} - 1)\Big], \forall n, k.$$

As stated in Eq.14, per-content services rates cumulatively equal the downlink capacity. This helps getting rid of the link service rate in the above expression and obtain that:

$$\sum_{k,a\in\Gamma^-(n)} \bar{\mu}^*_{k,a,n} = \sum_k \Big(\frac{\phi^*_k}{q_k}\Big)^{\alpha} \Big[\nu^{(2)*}_n + \nu^{(3)*}_{n,k}(2\bar{h}^*_{n,k} - 1)\Big] = \sum_{a\in\Gamma^-(n)} C_{n,a}, \forall n.$$

Then we substantiate the pivotal multiplier

$$\nu^{(2)*}_n = \Big[\sum_k \Big(\frac{\phi^*_k}{q_k}\Big)^{\alpha}\Big]^{-1} \mathcal{C}_n, \forall n,$$

$$\text{where } \mathcal{C}_n = \sum_{a\in\Gamma^-(n)} C_{n,a} - \sum_k \Big(\frac{\phi^*_k}{q_k}\Big)^{\alpha} \nu^{(3)*}_{n,k}(2\bar{h}^*_{n,k} - 1).$$

By contradiction, suppose $\nu^{(3)*}_{n,k} = 0$. It entails

$$\begin{aligned}\sum_{a\in\Gamma^-(n)} \bar{\mu}^*_{k,a,n} &= \frac{(\phi^*_k/q_k)^{\alpha}}{\sum_i (\phi^*_i/q_i)^{\alpha}} \mathcal{C}_n, \forall \alpha \geq 0 \\ &= \frac{\mathcal{C}_n}{|\mathcal{K}|}, \text{for } \alpha = 0 \text{ and } \alpha \to \infty.\end{aligned}$$

However, from Eq.B.1, $\forall \alpha \geq 0$, $\bar{h}^*_{n,1} = 1$ and $\bar{h}^*_{n,k} = 0, k \uparrow \infty$ yield $\epsilon(\alpha) > 0$. Consequently, as $\nu^{(3)*}_{n,k} = 0$ does not lead to a maximum, $\nu^{(3)*}_{n,k} > 0$ and $\bar{h}^*_{n,k} \in \{0, 1\}$. □