Implicit Coordination of Caches in Small Cell Networks under Unknown Popularity Profiles

Emilio Leonardi, Giovanni Neglia

Implicit Coordination of Caches in Small Cell Networks under Unknown Popularity Profiles

Emilio Leonardi† and Giovanni Neglia*
†Politecnico di Torino, Italy, leonardi@polito.it
*Université Côte d'Azur - Inria, France, giovanni.neglia@inria.fr

***Abstract*—We focus on a dense cellular network, in which a limited-size cache is available at every Base Station (BS). In order to optimize the overall performance of the system in such scenario, where a significant fraction of the users is covered by several BSs, a tight coordination among nearby caches is needed. To this end, this paper introduces a class of simple and fully distributed caching policies, which require neither direct communication among BSs, nor a priori knowledge of content popularity. Furthermore, we propose a novel approximate analytical methodology to assess the performance of interacting caches under such policies. Our approach builds upon the well known characteristic time approximation [1] and provides predictions that are surprisingly accurate (hardly distinguishable from the simulations) in most of the scenarios. Both synthetic and trace-driven results show that the our caching policies achieve excellent performance (in some cases provably optimal). They outperform state-of-the-art dynamic policies for interacting caches, and, in some cases, also the greedy content placement, which is known to be the best performing polynomial algorithm under static and perfectly-known content popularity profiles.**

I. Introduction

In the last years, with the advent and the proliferation of mobile devices (smart-phones, tablets), along with a constant increase of the overall traffic flowing over Internet, we have assisted to a radical shift of the traffic at the edge, from the wired/fixed segment of the network to the wireless/mobile segment. This trend is expected to continue and intensify in the next few years. According to CISCO forecasts [2] in the 5 years ranging from 2016 to 2021 traffic demand on the cellular network will approximately increase by a factor 8. Such traffic increase may pose a tremendous stress on the wireless infrastructure and can be satisfied only by densifying the cellular network and redesigning its infrastructure. To this end, the integration of caches at the edge of the cellular network can be effective to reduce the load on the back-haul links. Caches, indeed, by moving contents closer to the user, can effectively contribute to "localize" the traffic, and to achieve: i) the reduction of the load on the core network and back-haul links; ii) the reduction of the latency perceived by the user.

In this paper we focus our attention on a dense cellular network, where caches are placed at every Base Station (BS) and a significant fraction of the users can be served (is "covered") by two or more BSs (whose cells are said to "overlap"). In this context, an important open question is how to effectively coordinate different edge-caches, so to optimize the global performance (typically the hit ratio, i.e. the fraction of users' requests that are satisfied by local caches). Given the possibility of partial cell overlap, the cache coordination scheme should, indeed, reach an optimal trade-off between two somewhat conflicting targets: i) make top-popular contents available everywhere, so to maximize the population of users who can retrieve them from local caches, ii) diversify the contents stored at overlapping cells, so to maximize the number of contents available to users in the overlap. Optimal solutions can be easily figured out for the two extreme cases: when cells do not overlap, every cache should be filled with the most popular contents; when cells overlap completely, every cache should be filled with a different set of contents. In the most general case, however, finding the optimal content allocation strategy requires the solution of an NP-hard problem [3].

In this paper, we propose a class of fully distributed schemes to coordinate caches in cellular systems with partially overlapping cells, so to maximize the overall hit ratio. Our policies are very simple, and, differently from most of the previous work, do not require any a priori knowledge of content popularity. In addition, we propose a novel analytical approach to accurately evaluate the performance of such caching systems with limited computational complexity.

A. Related work

Due to the space constraints, we limit our description to the work that specifically addresses the caching problem in dense cellular networks.

To the best of our knowledge, the idea to coordinate content placement at caches located at close-by BSs

was first proposed in [4] and its extension [3] under the name of femto-caching. This work assumes that requests follow the Independent Reference Model (IRM) and geographical popularity profiles are available, i.e. content requests are independent and request rates are known for all the cell areas and their intersections. The optimal content placement to maximize the hit ratio has been formulated in terms of an NP-hard combinatorial problem. A greedy heuristic algorithm was then proposed and its performance analyzed. In particular the algorithm is shown to guarantee a $\frac{1}{2}$-approximation of the maximum hit ratio. In [5], the authors have generalized the approach of [4], [3], providing a formulation for the joint content-placement and user-association problem that maximizes the hit ratio. Efficient heuristic solutions have also been proposed. Authors of [6] have included the bandwidth costs in the formulation, and have proposed an on-line algorithm for the solution of the resulting problem. [7] considers the case when small cells can coordinate not just in terms of what to cache but also to perform Joint Transmission. In [8], instead, the authors have designed a distributed algorithm based on Gibbs sampling, which is shown to asymptotically converge to the optimal allocation. [9] revisits the optimal content placement problem within a stochastic geometry framework. Under the assumption that both base stations and users are points of two homogeneous spatial Poisson processes, it derives an elegant analytical characterization of the optimal policy and its performance. More recently, in [10] the authors have developed a few asynchronous distributed cooperative content placement algorithms with polynomial complexity and limited communication overhead (communication takes place only between overlapping cells), whose performance has been shown to be very good in most of the tested scenarios.

We would like to emphasize that all the previously proposed schemes, differently from ours, rely on the original assumption in [4] that geographical content popularity profiles are known by the system. Therefore we will refer to these policies as "informed" ones.

Reliable popularity estimates over small geographical areas may be very hard to obtain [11], because i) most of the contents are highly ephemeral, ii) few users are located in a small cell, and iii) users keep moving from one cell to another. On the contrary, policies like LRU and its variants (QLRU, 2LRU, ...) do not rely on popularity estimation—we call them "uninformed"—and are known to well behave under time-varying popularities. For this reason they are a de-facto standard in most of the deployed caching systems. [12] proposes a generalization of LRU to a dense cellular scenario. As above, a user at the intersection of multiple cells, can check the availability of the content at every covering cell and then download from one of it. The difference with respect to standard LRU is how cache states are updated. In particular, the authors of [12] consider two schemes: LRU-ONE and LRU-ALL. In LRU-ONE each user is assigned to a reference cell/cache and only the state of her reference cache is updated upon a hit or a miss, independently from which cache the content has been retrieved from.[1] In LRU-ALL the state of all the caches covering the user is updated. These policies do not require communication among caches. Moreover, their analysis is relatively easy because each cache can be studied as an isolated one. Unfortunately, these policies typically perform significantly worse than informed schemes (see for example the experiments in [10]).

B. Paper Contribution

This paper has four main contributions.

First, we propose in Sec. III a novel approximate analytical approach to study systems of interacting caches, under different caching policies. Our framework builds upon the well known characteristic time approximation [1] for individual caches, and, in most of the scenarios, provides predictions that are surprisingly accurate (practically indistinguishable from the simulations, as shown in Sec IV).

Second, we propose a class of simple and fully distributed "uniformed" schemes that effectively coordinate different caches in a dense cellular scenario in order to maximize the overall hit ratio of the caching system. Our schemes represent an enhancement of those proposed in [12]. As LRU-ONE and LRU-ALL, our policies neither rely on popularity estimation, nor require communication exchange among the BSs. The schemes achieve *implicit* coordination among the caches through specific cache update rules, which are driven by the users' requests. Differently from [12], our update rules tend to couple the states of different caches. Despite the additional complexity, we show that accurate analytical evaluation of the system is still possible through our approximate analytical approach.

Third, we rely on our analytical model to show that, under IRM, our policies can significantly outperform other uninformed policies like those in [12]. Moreover, the hit ratios are very close to those offered by the greedy scheme proposed in [4] under perfect knowledge

[1] The verbal description of the policy in [12] is a bit ambiguous, but the model equation shows that the state of a cache is updated by and only by the requests originated in the corresponding Voronoi cell, i.e. from the users closest to the cache.

of popularity profiles. More precisely, we can prove that, under some geometrical assumptions, a variant of QLRU asymptotically converges to the optimal static content allocation, while in more general scenarios, the same version of QLRU asymptotically achieves a locally optimal configuration.

Finally, in Sec. VI, we carry on simulations using a request trace from a major CDN provider and BS locations in Berlin, Germany. The simulations confirm qualitatively the model's results. Moreover, under the real request trace, our dynamic policies can sometimes outperform the greedy static allocation that knows in advance the future request rate. This happens even more often in the realistic case when future popularity needs to be estimated from past statistics. Overall, our results suggest that it is better to rely on uninformed caching schemes with smart update rules than on informed ones fed by estimated popularity profiles, partially contradicting some of the conclusions of [13].

II. Network Operation

We consider a set of B base stations (BSs) arbitrarily located in a given region $R \subseteq \mathbf{R}^2$, each equipped with a local cache. Our system operates as follows. When user u has a request for content f, it broadcasts an inquiry message to the set of BSs (I_u) it can communicate with. The subset ($J_{u,f}$) of those BSs that have the content f stored locally declare its availability to user u. If any local copy is available ($J_{u,f} \neq \emptyset$), the user sends an explicit request to download it to one of the BSs in $J_{u,f}$. Otherwise, the user sends the request to one of the BSs in I_u, which will need to retrieve it from the content provider.[2] Different user criteria can be defined to select the BS to download from; for the sake of simplicity, in this paper, we assume that the user selects uniformly at random one of them. However, the analysis developed in the next section extends naturally under general selection criteria. The selected BS serves the request. Furthermore, an opportunely defined subset of BSs in I_u updates its cache according to a local caching policy, like LRU, QLRU[3] or 2LRU,[4] etc. The most natural update rule is that only the cache serving the content updates its state independently from the identity of the user generating the specific request. We call this update rule *blind*. At the same time, it is possible to decouple content retrieval from cache state update as proposed in [12]. For example each user u may be statically associated to a given BS, whose state is updated upon every request from u independently from which BS has served the content. We refer to this update rule as *one*, because of the name of the corresponding policy proposed in [12] (LRU-ONE). Similarly, we indicate as *all* the update rule where all the BSs in I_u update their state upon a request from user u (as in LRU-ALL). These update rules can be freely combined with existing usual single cache policies, like QLRU, LRU, 2LRU, etc., leading then to schemes like LRU-ONE, QLRU-BLIND, 2LRU-ALL, etc., with obvious interpretation of the names.

The analytical framework presented in the next section allows us to study a larger set of update rules, where the update can in general depend on the identity of the user as well as on the current set $J_{u,f}$ of caches from which u can retrieve the content. When coupled with local caching policies, these update rules do not require any explicit information exchange among the caches, but they can be implemented by simply piggybacking the required information ($J_{u,f}$) to user u's request. In particular, in what follows, we will consider the *lazy* update rule, according to which

1) only the cache serving the content may update its state,
2) but it does only if no other cache could have served the content to the user (i.e. only if $|J_{u,f}| \leq 1$).

This rule requires only an additional bit to be transmitted from the user to the cache. We are going to show in Sec. V that such bit is sufficient to achieve a high level of coordination among different caches and, therefore, a high hit ratio. Because no communication among BSs is required, we talk about *implicit coordination*. For the moment, the reader should not be worried if he/she finds the rationale behind *lazy* obscure and can regard *lazy* as a specific update rule among many others possible. Table I summarises the main notation used in this paper.

III. Model

We assume that mobile users are spread over the region R according to a Poisson point process with density $\mu()$, so that $\mu(A)$ denotes the expected number of users in a given area $A \subseteq R$. Users generate independent requests for F possible contents. In particular, a given user requests content f according to a Poisson process with rate λ_f. It follows that the aggregate request process

[2] This two-step procedure introduces some additional delay, but this is inevitable in any femtocaching scheme where the BSs need to coordinate to serve the content.

[3] In the case of QLRU, the cache will move the content requested to the front of the queue upon a hit and will store the content at the front of the cache with probability q upon a miss. LRU is a QLRU policy with $q = 1$.

[4] 2LRU uses internally two LRU caches: one for the metadata, and the other for the actual contents (for this reason we say that 2LRU is a two-stage cache). Upon a miss, a content is stored in the second cache only if its metadata are already present in the first cache. See [14] for a more detailed description.

TABLE I
SUMMARY OF THE MAIN NOTATION

Symbol	Explanation
t	time
u	generic user
f	generic content
b	generic cell
F	number of files
B	number of base stations
C	cache size
I_u	set of BSs communicating with u
$J_{u,f}$	set of BSs able to provide f to u
S_b	surface of cell b
$\mu(A)$	expected number of users in region A
$\Lambda_f(A)$	content f request rate from region A
$\mathbf{X}_f$	configuration of content f in caches
$x_f^{(b)}$	component b of $\mathbf{X}_f$: $x_f^{(b)} \in \{0,1\}$
$\mathbf{x}_f^{(-b)}$	configuration of content f in all caches but b
$T_c^{(b)}$	characteristic time at cache b
$T_{S,f}^{(b)}$	sojourn time of content f in cache b
$\nu_f^{(b)}$	transition rate $1 \to 0$ for x_f^b given $\mathbf{x}_f^{(-b)}$
$\alpha_f^{(b)}$	transition rate $0 \to 1$ for x_f^b given $\mathbf{x}_f^{(-b)}$
$d(s,A)$	distance between point s and region A

for content f from all the users located in A is also a Poisson process with rate $\Lambda_f(A) = \mu(A)\lambda_f$. For the sake of presentation, in what follows we will consider that users' density is constant over the region, so that $\mu(A)$ is simply proportional to the surface of A, but our results can be easily generalized. Our analysis can also be extended to a more complex content request model that takes into account temporal locality [15] as we discuss in Sec. III-F. Contents (i.e. caching units) are assumed to have the same size. This assumption can be justified in light of the fact that often contents correspond to chunks in which larger files are broken. In any case, it is possible to extend the model, and most of the analytical results below, to the case of heterogeneous size contents.[5] For the sake of simplicity, we assume that each cache is able to store C contents. Finally, let S_b denote the coverage area of BS b.

In what follows, we first present some key observations for an isolated cache, and then we extend our investigation to interacting caches when cells overlap. We will first consider the more natural *blind* update rule, according to which any request served by a BS triggers a corresponding cache state update. We will then discuss how to extend the model to other update rules in Sec. III-D.

[5] Similar results for QLRU-LAZY in Sec. V-B hold if we let the parameter q be inversely proportional to the content size as done in [16].

A. A single cell in isolation

We start considering a single BS, say it b, with cell size S_b. The request rate per content f is then $\Lambda_f^{(b)}(S_b) = \mu(S_b)\lambda_f$. We omit in what follows the dependence on S_b.

Our analysis relies on the now standard cache characteristic time approximation (CTA) for a cache in isolation, which is known to be one of the most effective approximate approaches for analysis of caching systems.[6] CTA was first introduced (and analytically justified) in [18] and later rediscovered in [1]. It was originally proposed for LRU under the IRM request process, and it has been later extended to different caching policies and different requests processes [14], [19]. The characteristic time T_c is the time a given content spends in the cache since its insertion until its eviction in absence of any request for it. In general, this time depends in a complex way from the dynamics of other contents requests. Instead, the CTA assumes that T_c is a random variable independent from other contents dynamics and with an assigned distribution (the same for every content). This assumption makes it possible to decouple the dynamics of the different contents: upon a miss for content f, the content is retrieved and a timer with random value T_c is generated. When the timer expires, the content is evicted from the cache. Cache policies differ for i) the distribution of T_c and ii) what happens to the timer upon a hit. For example, T_c is a constant under LRU, QLRU, 2LRU and FIFO and exponentially distributed under RANDOM. Upon a hit, the timer is renewed under LRU, QLRU and 2LRU, but not under FIFO or RANDOM. Despite its simplicity, CTA was shown to provide asymptotically exact predictions for a single LRU cache under IRM as the cache size grows large [18], [20], [21].

What is important for our purposes is that, once inserted in the cache, a given content f will sojourn in the cache for a random amount of time $T_{S,f}$, that can be characterized for the different policies. In particular, if the timer is not renewed upon a hit (as for FIFO and RANDOM), it holds:

$$T_{S,f}^{(b)} = T_c^{(b)},$$

while if the timer is renewed, it holds:

$$T_{S,f}^{(b)} = \sum_{k=1}^{M} Y_k + T_c^{(b)},$$

where $M \in \{0,1,\dots\}$ is the number of consecutive hits preceding a miss and Y_k is the time interval be-

[6] Unfortunately, the computational cost to exactly analyse even a single LRU (Least Recently Used) cache, grows exponentially with both the cache size and the number of contents [17].

tween the k-th hit and the previous content request. For example, in the case of LRU and QLRU, M is distributed as a geometric random variable with parameter $p = 1 - e^{-\Lambda_f^{(b)} T_c^{(b)}}$, and $\{Y_k\}$ are i.i.d. truncated exponential random variables over the interval $[0, T_c^{(b)}]$.

We denote by $1/\nu_f^{(b)}$ the expected value of $T_{S,f}^{(b)}$, that is a function of the request arrival rate $\Lambda_f^{(b)}$.

$$\frac{1}{\nu_f^{(b)}\left(\Lambda_f^{(b)}\right)} \triangleq \mathbf{E}[T_{S,f}^{(b)}]. \tag{1}$$

For example it holds:

$$\text{FIFO, RANDOM:} \quad \nu_f^{(b)}\left(\Lambda_f^{(b)}\right) = 1/T_c^{(b)}$$

$$\text{LRU, QLRU:} \quad \nu_f^{(b)}\left(\Lambda_f^{(b)}\right) = \frac{\Lambda_f^{(b)}}{e^{\Lambda_f^{(b)} T_c^{(b)}} - 1}.$$

where the last expression can be obtained through standard renewal arguments (see [22]).

Let $X_f^{(b)}(t)$ be the process indicating whether content f is in the cache b at time t. For single-stage caching policies, such as FIFO, RANDOM, LRU and QLRU, $X_f^{(b)}(t)$ is an ON/OFF renewal process with ON period distributed as $T_{S,f}^{(b)}$ and OFF period distributed exponentially with mean value $1/\alpha_f^{(b)}$, where $\alpha_f^{(b)} = \Lambda_f^{(b)}$ for FIFO and RANDOM, and $\alpha_f^{(b)} = q\Lambda_f^{(b)}$ for QLRU. The process $X_f^{(b)}(t)$ can also be considered as the busy server indicator of an $M/G/1/0$ queue with service time distributed as $T_{S,f}^{(b)}$.[7] This observation is important because the stationary distribution of $M/G/n/0$ queues depends on the service time only through its mean [23]. As a consequence, for any metric depending only on the stationary distribution, an $M/G/1/0$ queue is equivalent to an $M/M/1/0$ queue with service time exponentially distributed with the same service rate $\nu_f^{(b)}$. In particular, the stationary occupancy probability $h_f^{(b)} = \Pr\{X_f^{(b)}(t) = 1\}$ is simply $h_f^{(b)} = \frac{\alpha_f^{(b)}/\nu_f^{(b)}}{\alpha_f^{(b)}/\nu_f^{(b)}+1}$. Under CTA, the characteristic time T_c can then be obtained by imposing that

$$\sum_{f=1}^{F} h_f^{(b)} = C, \tag{2}$$

for example using the bisection method.

The possibility of representing a cache in isolation with an $M/M/1/0$ queue, i.e., as a simple continuous time Markov chain, does not provide particular advantages in this simple scenario, but it allows us to accurately study the more complex case when cells overlap, and users' request may be served by multiple caches.

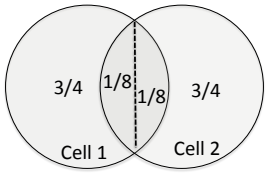


Fig. 1. Two overlapping cells each with unit surface. The area of the overlapping area is $1/4$.

B. Overlapping cells

We consider now the case when B cells may overlap. Let $X_f^{(b)}(t)$ indicate whether the BS b stores at time t a copy of content f and $\mathbf{X}_f(t) = \left(X_f^{(1)}(t), \ldots X_f^{(B)}(t)\right)$ be the vector showing where the content is placed within the network. In this case the request rate seen by any BS, say it BS b, depends on the availability of the content at the neighbouring BSs, i.e. $\Lambda_f^{(b)} = \Lambda_f^{(b)}(\mathbf{X}_f(t))$. For example, with reference to the Fig. 1, if $\lambda_f = 1$, BS 1 experiences a request rate for content f equal to i) 1 if it is the only BS to store the content, ii) $7/8$ if both BSs store the content or none of them does, iii) $3/4$ if only BS 2 stores the content.

Our analysis of this system is based on the following approximation:[8]

A1 The stochastic process $\mathbf{X}_f(t)$ is a continuous-time Markov chain. For each f and b the transition rate $\nu_f^{(b)}$ from state $\mathbf{X}_f(t) = (x_f^{(b)} = 1, \mathbf{x}_f^{(-b)})$ to $(x_f^{(b)} = 0, \mathbf{x}_f^{(-b)})$ is given by (1) with $\Lambda_f^{(b)}$ replaced by $\Lambda_f^{(b)}(\mathbf{X}_f(t))$.

Before discussing the quality of approximation A1, let us first describe how it allows us to study the cache system. For a given initial guess of the characteristic times at all the B caches, we determine the stationary distribution of the Markov Chains (MCs) $\mathbf{X}_f(t)$. We then compute the expected buffer occupancy at each cache and check if the set of constraints (2) is satisfied. We then iteratively modify the vector of characteristic times by reducing (/increasing) the value for those caches where the expected buffer occupancy is above (/below) C. Once the iterative procedure on vector of characteristic times has reached convergence, we compute the hit ratios for each content at each cache.

A1 envisages to replace the original stochastic process, whose analysis appears prohibitive, with a (simpler) MC.

[7] Under CTA a cache with capacity C becomes indeed equivalent to a set of C parallel independent $M/G/1/0$ queues, one for each content.

[8] For any vector $\mathbf{x}$, we denote by $\mathbf{x}^{(-b)}$ the subvector of $\mathbf{x}$ including all the components but the b-th one and we can write $\mathbf{x}$ as $(x^{(b)}, \mathbf{x}^{(-b)})$.

This has no impact on any system metric that depends only on the stationary distribution in the following cases:

1) isolated caches (as we have shown in Sec. III-A),
2) caches using RANDOM policy, because the corresponding sojourn times coincide with the characteristic times and are exponentially distributed, hence A1 is not an approximation,
3) caches using FIFO policy under the additional condition in Proposition III.1 below.

In all these cases CTA is the only approximation having an impact on the accuracy of model results. In the most general case, however, A1 introduces an additional approximation. However our numerical evaluation shows that our approach provides very accurate results in all the scenarios we tested.

We end this section by detailing the insensitivity result for a system of FIFO caches.

Proposition III.1. *For* FIFO*, the probability of being in state* $\mathbf{x}_f$ *is insensitive to the distribution of the sojourn times* $T_{S,f}^{(b)}$ *as far as the Markov chain* $\mathbf{X}_f(t)$ *in approximation A1 is reversible.*

The proof of proposition III.1 is in Appendix I and relies on some insensitivity results for Generalized Semi Markov Processes. The reversibility hypothesis is for example satisfied for the cell trefoil topology considered in Sec. IV when users' density is constant.

C. Model complexity

Note that, in general, the number of states of the Markov Chain describing the dynamics of $\mathbf{X}_f(t)$ grows exponentially with the number of cells B (actually, it is equal to 2^B), therefore modeling scenarios with a large number of cells becomes challenging and requires the adoption of efficient approximate techniques for the solution of probabilistic graphical methods [24]. However, scenarios with up to 10-12 cells can be efficiently modeled. Furthermore, when the geometry exhibits some symmetry, some state aggregation becomes possible. For example, in the cell trefoil topology presented in the next section, the evolution of $\mathbf{X}_f(t)$ can be represented by a reversible birth-and-death Markov Chain with $(B+1)$ states ($\ll 2^B$).

D. Different Update rules

In presenting the model above, we have referred to the simple *blind* update rule. Our modeling framework, however, can easily accommodate other update rules. For example for *one*, if the reference BS is the closest one, we should set $\Lambda_f^{(b)} = \lambda_f \mu\left(\{\mathbf{s} \in R; d(\mathbf{s}, S_b) \leq d(\mathbf{s}, S_{b'}), \forall b'\}\right)$, where $d(\mathbf{s}, A)$ denotes the distance between the point $\mathbf{s}$ and the set A. On the contrary, for *all*, any request that could be served by the base station is taken into account, i.e. $\Lambda_f^{(b)} = \lambda_f \mu\left(S_b\right)$. Finally, for *lazy* we have:

$$\Lambda_f^{(b)}(\mathbf{X}_f) = \lambda_f \mu \left(S_b \setminus \bigcup_{b' | X_f^{(b')}=1} S_{b'} \right), \qquad (3)$$

i.e. only requests coming from areas that cannot be served from any other cache, affect the cache state. For example, with reference to Fig. 1, assuming $\lambda_f = 1$, the request rate that contributes to update cache 1 status is $3/4$ when content f is stored also at cache 2.

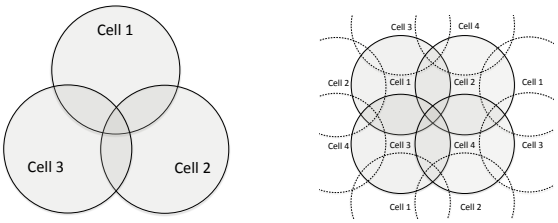
As we are going to discuss in Sec. V, the update rules have a significant impact on the performance and in particular the lazy policies often outperform the others. Because our analysis will rely on the model described in this section, we first present in Sec. IV some validation results to convince the reader of its accuracy.

E. Extension to multistage caching policies: κLRU

The previous model can be extended to 2LRU (and κLRU) by following the approach proposed in [14]. In particular dynamics of the two stages can be represented by two separate continuous time MCs whose states $\mathbf{X}_f^{(1)}(t)$ and $\mathbf{X}_f^{(2)}(t)$ correspond to the configuration of content f at time t in the system of virtual caches and physical caches, respectively. The dynamics of the system of virtual caches at the first stage $\mathbf{X}_f^{(1)}(t)$ are not impacted by the presence of the second stage and perfectly emulate the dynamics of LRU caches; therefore we model them by using the same MC as for LRU. On the contrary, dynamics at the second stage depend on the first stage state. In particular content f is inserted in the physical cache at the second stage upon a miss, only if the incoming request finds the content metadata within the first stage cache. Therefore the transition rate from state $\mathbf{X}_f^{(2)}(t) = (x_f^{(b,2)} = 0, \mathbf{x}_f^{(-b,2)})$ to $(x_f^{(b,2)} = 1, \mathbf{x}_f^{(-b,2)})$ is given by $\lambda_f^{(b,2)} = \Lambda_f^{(b)}(\mathbf{X}_f^{(2)}(t)) h_f^{(b,1)}$, where $h_f^{(b,1)}$ represents the probability that content f metadata is stored at the first stage. Along the same lines the model can be easily extended to κLRU for $k > 2$.

F. How to account for temporal locality

Following the approach proposed in [15], [19], we model the request process of every content f as a Markov Modulated Poisson Process (MMPP), whose modulating MC is a simple ON-OFF MC. Now focusing, first, on a single cell scenario, we denote by $\Lambda_f^{(b)}$ the aggregate



(a) (b)

Fig. 2. (a): a trefoil. (b): a two-by-two cell torus.

arrival rate of content f at BS b during ON periods. The arrival rate of content f is, instead, null during OFF periods. Let T_f^{ON} and T_f^{OFF} denote the average sojourn times in state ON and OFF, respectively.[9] The idea behind this model is that each content has a finite lifetime with mean T_f^{ON} and after a random time with mean T_f^{OFF}, a new content with the same popularity arrives in the system. For convenience this new content is denoted by the same label f (see [15], [19] for a deeper discussion). We can model the dynamics of content f in the cache as an MMPP/M/1/0 queue with state-dependent service rate. In particular service rates upon ON ($\nu_f^{(b,ON)}$) are computed according to (1). Service rates on state OFF are simply set to $\nu_f^{(b,OFF)} = \frac{1}{T_c^{(b)}}$, as result of the application of (1) when the arrival rate of content-f requests tends to 0.

The extension to the case of multiple overlapping cells can be carried out along the same lines of Section III-B, (i.e. by applying approximation A1). As in [19], the ON-OFF processes governing content-f request rate at different cells are assumed to be perfectly synchronized (i.e., a unique underlying ON-OFF Markov Chain determines content-f request rate at every cell). The resulting stochastic process $\mathbf{X}_f(t)$ is a continuous-time Markov Chain with 2^{B+1} states.

IV. Model validation

In this section we validate our model by comparing its prediction against simulation results for two different topologies. Our trace-driven simulator developed in Python reproduces the exact dynamics of the caching system, and therefore can be used to test the impact of model assumptions (CTA and A1) on the accuracy of results in different traffic scenarios. We start introducing a topology which exhibits a complete cell symmetry (i.e. the hit rate of any allocation is invariant under cell-label permutations). In such a case, $\mathbf{X}_f(t)$ turns out to be a reversible Markov Chain. Fig. 2 (a) shows an example for $B = 3$. Generalizations for $B > 3$ can be defined in a B dimensional Euclidean-space by considering B hyperspheres centered at the vertices of a regular simplex, but also on the plane if users' density is not homogeneous. We refer to this topology as the *trefoil*. Then we consider a *torus topology* in which the base stations are disposed according to a regular grid on a torus as in Fig. 2 (b). For simplicity, in what follows, we assume that all the cells have the same size and a circular shape.

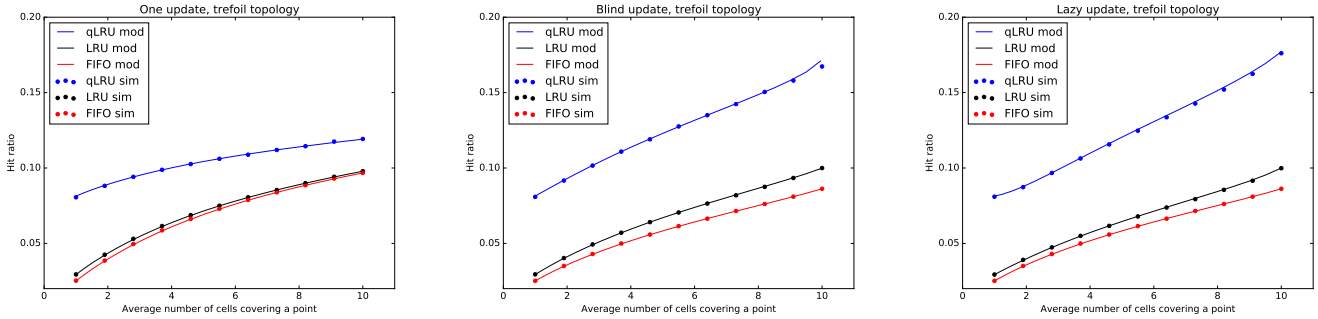
Users are uniformly distributed over the plane and they request contents from a catalogue of $F = 10^6$ contents whose popularity is distributed according to a Zipf's law with exponent $s = 0.8$. Each BS can store up to $C = 100$ contents. We have also performed experiments with $C = 1000$ and $s = 0.7$, but the conclusions are the same, so we omit them due to space constraints.

In Fig. 3 we show the global hit ratio for different values of cell overlap in a trefoil topology with 10 cells. The overlap is expressed in terms of the expected number of caches a random user could download the content from. The subfigure (a) shows the corresponding curves for FIFO-ONE and QLRU-ONE with $q = 0.01$ and with $q = 1$, which coincides with LRU-ONE. The other subfigures are relative to the update rules *blind* and *lazy*.[10] FIFO-BLIND and FIFO-LAZY coincide because in any case FIFO does not update the cache status upon a hit. The curves show an almost perfect matching between the results of the model described in Sec. III and those of simulation. Figure 4 confirms the accuracy of the model also for the torus topology with 9 cells. Every model point has requested less than 3 seconds of CPU-time on a INTEL Pentium G3420 @3.2Ghz for the cell trefoil topology, and less than 5 minutes for the torus.

Finally, Fig. 5 shows that the model is also accurate when the request process differs from IRM. The curves have been obtained for a trefoil topology under the *lazy* update rule and the ON-OFF traffic model described and studied in Sec. III-F. In the figure we also show some results for 2LRU-LAZY. As QLRU, upon a miss, 2LRU prefilters the contents to be stored in the cache. QLRU does it probabilistically, while 2LRU exploits another LRU cache for the metadata. Under IRM, their performance are qualitatively similar, but 2LRU is known to be more reactive and then better performing when the request process exhibits significant temporal locality [14]. Our results in Fig. 5 confirm this finding. In particular, as q decreases, the performance of QLRU first improves (compare $q = 0.01$ with $q = 1$)

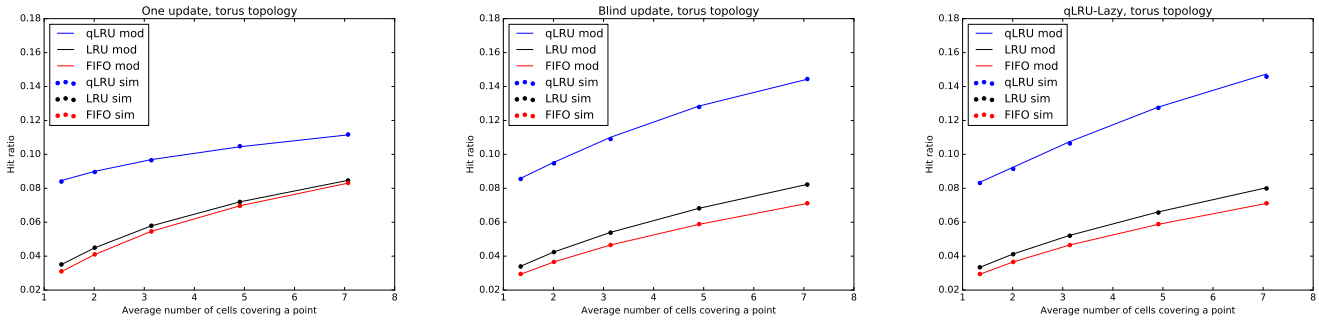
[9] Sojourn times in both states are exponentially distributed.

[10] We do not show results for RANDOM or the update rule *all*. RANDOM is practically indistinguishable from FIFO and *all* was shown to have worse performance than *one* for IRM traffic already in [12].



(a) One (b) Blind (c) Lazy

Fig. 3. Comparison between model predictions and simulations; trefoil topology with 10 cells; $C = 100$; IRM traffic model with $\alpha = 0.8$; QLRU employs $q = 0.01$.



(a) One (b) Blind (c) Lazy

Fig. 4. Comparison between model predictions and simulations; torus topology with 9 cells; $C = 100$; IRM traffic model with $\alpha = 0.8$; QLRU employs $q = 0.01$.

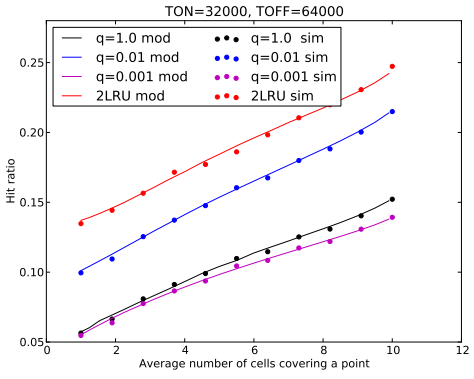


Fig. 5. Trefoil: results for the *lazy* update rule and an ON-OFF request process with $\mathbf{E}[T_f^{ON}] = 3.2{*}10^4$ and $\mathbf{E}[T_f^{OFF}] = 6.4{*}10^4$. The cell request rate for the most popular content is $\Lambda = 1.3$.

because QLRU's probabilistic admission rule filters the unpopular content, and then worsens (see the curve for $q = 0.001$) when QLRU dynamics' timescale becomes comparable to T_{ON}.

V. The Lazy Update Rule

Even if the focus of the previous section has mainly been on the validation of our model, the reader may have observed by looking at Figures 3 and 4 that the update rule *lazy* performs significantly better than *one* and (to a lesser extent) *blind*, especially when the cellular network is particularly dense. This improvement comes at the cost of a minimal communication overhead: an additional bit to be piggybacked into every user's request to indicate whether the content is available at some of the other cells covering the user. In this section we use our model to further investigate the performance of the *lazy* update rule.

First, we present in Fig. 6 some results for QLRU coupled with the different update rules. The curves show the hit ratio versus the parameter q achieved by the different policies. The topology is a trefoil with 10 cells. Results are reported for two values of cell overlap, corresponding to the cases where a user is covered on average by 5 and 10 BSs. As a reference, also the optimal achievable hit ratio is shown by the two horizontal green lines (in this particular scenario,

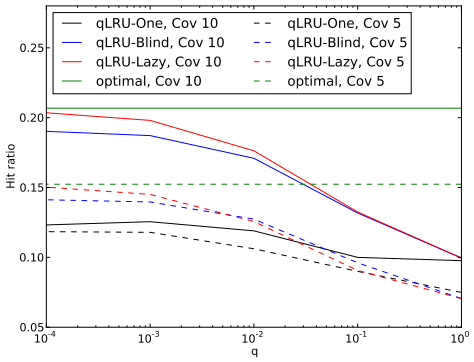


Fig. 6. Performance of QLRU coupled with different update rules in a trefoil network (model results).

the optimal allocation can be obtained by applying the greedy algorithm described below in Sec. V-A). QLRU-LAZY significantly outperforms QLRU-ONE for small values of q, with relative gain that can be as high as 25% for the 5-coverage and 65% for the 10-coverage. The improvement with respect to *blind* is smaller, but what is remarkable is that QLRU-LAZY appears to be able to asymptotically approach the performance of the optimal allocation. In the following we will prove that i) this is indeed the case for the trefoil topology and ii) QLRU-LAZY achieves a locally optimal allocation in a general scenario. For a single cache, it has already been proven that QLRU asymptotically maximizes the hit ratio when q converges to 0 (see [14] for the case of uniform content size contents and [25] for the case of heterogeneous size), but, to the best of our knowledge, no optimality results are available for a multi-cache scenario as the one we are considering. Before proving optimality, we discuss what is the optimal allocation and we provide some intuitive explanation about *lazy* good performance.

A. Optimal content allocation and a new point of view on lazy

If content popularities are known and stationary, one can allocate, once and for all, contents to caches in order to maximize the global hit ratio. Formally, the following integer maximization problem can be defined:

$$\text{maximize} \quad \sum_{f=1}^{F} \lambda_f \mu \left(\bigcup_{b|x_f^{(b)}=1} S_b \right) \tag{4}$$

$$\text{subject to} \quad \sum_{f=1}^{F} x_f^{(b)} = C \quad \forall b = 1, \dots B,$$

$$x_f^{(b)} \in \{0,1\} \quad \forall f = 1, \dots F, \quad \forall b = 1, \dots B.$$

Carrying on an analysis similar to that in [3], it is possible to show that this problem i) is NP-hard (e.g. through a reduction to the 2-Disjoint Set Cover Problem), ii) can be formulated as the maximization of a monotone sub-modular set function with matroid constraints. It follows that the associated greedy algorithm provides a $1/2$-approximation for problem (4).

Let us consider how the greedy algorithm operates. Let $\mathbf{X}(l-1) \in \{0,1\}^{B \times F}$ describe the allocation at the $(l-1)$-th step of the greedy algorithm, i.e. the matrix element $(\mathbf{X}(l-1))_{f,b} = x_f^{(b)}(l-1)$ indicates if at step $l-1$ the algorithm places content f at cache b. At step l, the greedy algorithm computes for each content f and each cache b the marginal improvement for the global hit ratio to store a copy of f at cache b, given the current allocation $\mathbf{X}(l-1)$, that is

$$\lambda_f \mu \left(S_b \setminus \bigcup_{b'|x_f^{(b')}(l-1)=1} S_{b'} \right) \tag{5}$$

The pair (f_l, b_l) leading to the largest hit ratio increase is then selected and the allocation is updated by setting $x_{f_l}^{(b_l)} = 1$. The procedure is iterated until all the caches are full.

We observe that (5) is exactly the request rate that drives the dynamics of QLRU-LAZY in state $\mathbf{X}_f(l-1)$, as indicated in (3). Upon a miss for content f, QLRU-LAZY inserts it with a probability that is proportional to the marginal increase of the global hit ratio provided by adding the additional copy of content f. This introduces a stochastic drift toward local maxima of the hit ratio. As we said above, when q vanishes, it is known that an isolated QLRU cache tends to store deterministically the top popular contents, then one can expect each QLRU-LAZY cache to store the contents with the largest marginal request rate given the current allocation at the other caches. Therefore, it seems licit to conjecture that a system of QLRU-LAZY caches asymptotically converges at least to a local maximum for the hit ratio (the objective function in (4)). Section V-C shows that this is indeed the case. Before moving to that result, we show that in particular QLRU-LAZY achieves the maximum hit ratio in a trefoil topology.

B. In a trefoil topology QLRU-LAZY achieves the global maximum hit ratio

Now, we formalize the previous arguments, showing that as q tends to 0, QLRU-LAZY content allocation converges to an optimal configuration in which the set of contents maximizing the global hit ratio is stored at the caches. This result holds for the trefoil topology under our model.

We recall that that the trefoil topology exhibits a complete cell symmetry and that the hit ratio of any

allocation is invariant under cell label permutations. A consequence is that the hit ratio depends only on the number of copies of each file that are stored in the network, while it does not depend on where they are stored as far as we avoid to place multiple copies of the same file in the same cache, that is obviously unhelpful. It is possible then to describe a possible solution simply as an F-dimensional vector $\mathbf{k} = (k_1, k_2, \dots k_F)$, where k_f denotes the number of copies of content f. Under QLRU-LAZY we denote by $\pi(q, \mathbf{k})$, the stationary probability that the system is in a state with allocation $\mathbf{k}$.

The optimality result follows from combining the two following propositions (whose complete proofs are in Appendix II):

Proposition V.1. *In a trefoil topology, an allocation of the greedy algorithm for Problem* (4) *is optimal.*

The proof relies on mapping problem (4) to a knapsack problem with $F \times C$ objects with unit size for which the greedy algorithm is optimal.

We observe that for generic values of the parameters, all the marginal improvements considered by the greedy algorithm are different and then the greedy algorithm admits a unique possible output (apart from BSs label permutations).

Proposition V.2. *Consider a trefoil topology and assume there is unique possible output of the greedy algorithm, denoted as* $\mathbf{k}^* = (k_1^*, k_2^*, \dots k_F^*)$. *Then, under the approximate model in Sec III, a system of* QLRU-LAZY *caches asymptotically converges to* $\mathbf{k}^*$ *when* q *vanishes in the sense that*

$$\lim_{q \to 0} \pi(q, \mathbf{k}^*) = 1.$$

In order to prove this result, we write down the explicit stationary probability for the system, taking advantage of the fact that the MC is reversible, and we study its limit. In conclusion the greedy algorithm and QLRU-LAZY are equivalent in the case of trefoil topology.

C. QLRU-LAZY *achieves a local maximum hit ratio*

We say that a caching configuration $\mathcal{C}$ is locally optimal if it provides the highest aggregate hit rate among all the caching configurations which can be obtained from $\mathcal{C}$ by replacing one content in one of the caches.

Proposition V.3. *A spatial network of* QLRU-LAZY *caches asymptotically achieves a locally-optimal caching configuration when* q *vanishes.*[11]

[11]In the most general case, the adoption of different parameters q is required at different cells for the implementation of the QLRU policy.

TABLE II
TRACE: BASIC INFORMATION

Time span	5 days
Number of requests received	$4 \cdot 10^8$
Number of distinct objects	$13 \cdot 10^6$

The proof is in Appendix II-C. In this general case the difficulty of proving the assertion stems from the fact that the MC representing content dynamics is not anymore reversible, and it is then difficult to derive an analytical expression for its steady state distribution. Instead, our proof relies on results for regular perturbations of Markov chains [26].

The analytical results in this section justify why for small, but strictly positive, values of q, QLRU-LAZY performs better than QLRU-BLIND and QLRU-ONE. More in general, what seems fundamental to approach the maximum hit ratio is the coupling of the *lazy* update rule, that reacts to the "right marginal benefit" for problem (4), with a caching policy that is effective to store the most popular contents. QLRU is one of them, 2LRU is another option. Moreover, 2LRU has been shown to react faster to popularity changes. For this reason, in the next section we also include results for 2LRU-LAZY. At last we wish to remark that the (static) cache configuration selected by greedy algorithm is in general not locally optimal, as a consequence of the greedy nature of the algorithm and the fact that marginal gains at a cell change during the execution of the algorithm (since they depend on the configuration of neighbouring cells).

VI. PERFORMANCE IN A REALISTIC DEPLOYMENT

In this section we evaluate the performance of the *lazy* update rule in a more realistic scenario. To this purpose, we have extracted the positions of 10 T-Mobile BSs in Berlin from the dataset in [27] and we use a real content request trace from Akamai Content Delivery Network [16]. The actual identity of the users and of the requested objects was obfuscated. The BS locations are indicated in Fig. 7. We refer to this topology simply as the Berlin topology. The trace includes 400 million requests issued over 5 days from users in the same geographical zone for a total of 13 million unique contents. In our simulations we randomly assign the requests to the users who are uniformly spread over the area.

Figure 8 compares the performance of different caching policies in this scenario, when the transmission range of the BSs varies from 25 to 250 meters and correspondingly a user is covered on average by 1.1 up

Fig. 7. T-Mobile BS configuration in Berlin.

to 9.4 BSs. We observe that the *lazy* update rule still outperforms *one* and *blind* when coupled with QLRU or 2LRU. Moreover, for the higher density scenarios, 2LRU-LAZY, 2LRU-BLIND, QLRU-LAZY and (to a minor extent) QLRU-BLIND outperform the static allocation that has been obtained by the greedy algorithm assuming known the request rates of each content over the future 5 days. While we recall that the greedy algorithm provides only a $1/2$-approximation of the optimal allocation for problem (4), we highlight that this apparently surprising result is most likely to be due to the non-stationarity of the request process. In this case an uninformed dynamic policy (like QLRU or 2LRU) can outperform an informed static one, by dynamically adapting content allocation in caches to the short-term request rate of contents.

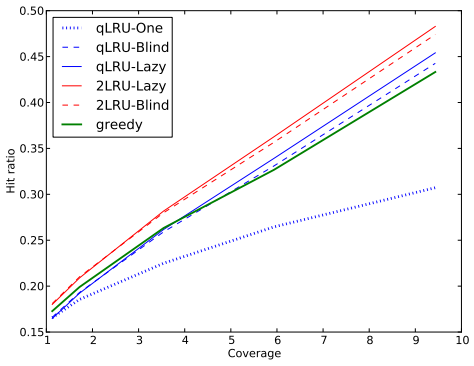


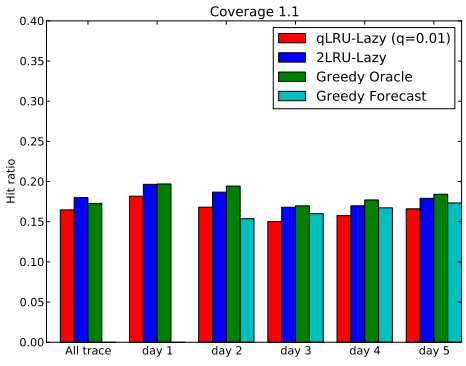
Fig. 8. Berlin topology and real CDN request trace. QLRU employs $q = 0.01$.

In order to deepen the comparison between uninformed and informed policies, we have considered the operation scenario that is usually suggested from supporters of informed policies (see e.g. [3]): the optimal allocation is computed the day ahead and contents are pushed to the caches during the night when the network is unloaded. Figure 9 shows then the performance for two coverage values (1.1 and 5.9) on a daily basis as well as for the whole 5 days. In this case, the oracle greedy algorithm computes a static content allocation each night knowing exactly the future request rates for the following day. Instead, the forecast greedy algorithm uses the request rates seen during the current day as an estimation for the following one. The oracle greedy benefits from the knowledge of the request rates over a single day: it can now correctly identify and store those contents that are going to be popular the following day, but are not so popular over the whole trace. For this reason, it outperforms the greedy scheme that receives as input the average rates over the whole 5-day period. When cells have limited overlap (Fig. 9 (a)), the oracle greedy algorithm still outperforms the dynamic policies, but 2LRU-LAZY is very close to it. Interestingly, in the higher density setting (Fig. 9 (b)), this advantage disappears. The performance of the 2LRU-LAZY allocation becomes preferable than both oracle greedy (with daily rates), and QLRU-LAZY. Temporal locality appears to have a larger impact in high density scenarios!

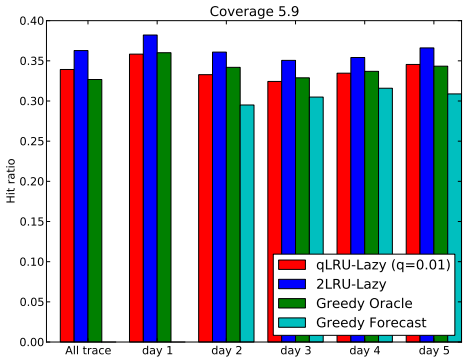
At last we wish to remark that the allocation of the oracle greedy algorithm is an ideal one, because it assumes the future request rates to be known. A practical algorithm will be necessarily based on some estimates such as the forecast greedy. Our results show a significant performance loss due to this incorrect input (as already observed in [28]). Both 2LRU-LAZY and QLRU-LAZY perform significantly better than forecast greedy (2LRU-LAZY guarantees between 10% and 20% improvement). Interestingly, our results contradict one of the conclusions in [13], i.e. that at the BS level reactive caching policies would not be efficient because the content request rate is too low, and content prefetching would perform better. We observe that [13] considers a single BS scenario and that perfect popularity knowledge is available. We have performed some additional simulations considering the current typical request rate at a BS as identified in [13] and we still observe qualitatively the same behaviour illustrated in Fig. 9. These additional experiments are described in Appendix III. Moreover, data traffic rate in cellular networks is constantly increasing and this improves the performance of reactive policies (but not of prefetching) as already observed in [13].

VII. CONCLUSIONS

In this paper, we have shown that "uniformed" schemes can effectively implicitly coordinate different caches in dense cellular systems when smart (but simple) update policies like *lazy* are used. Indeed we show that they can achieve a performance, which is comparable to



(a) Coverage 1.1



(b) Coverage 5.9

Fig. 9. Berlin topology and real CDN request trace. Comparison of the different policies over the whole trace and for each of the 5 days. QLRU employs $q = 0.01$.

that of the best "informed" schemes in static traffic scenarios. Moreover, "uniformed" schemes better adapt to dynamic scenarios, often outperforming implementable "informed" schemes. For once, then, sloth is not the key to poverty, not at least to poor performance.

We have also proposed a new approximate analytical framework to assess the performance of "uniformed" schemes. The predictions of our model are extremely accurate (hardly distinguishable from Monte Carlo simulations in most cases).

This work was partly funded by the French Government (National Research Agency, ANR) through the "Investments for the Future" Program reference #ANR-11-LABX-0031-01.

REFERENCES

[1] H. Che, Y. Tung, and Z. Wang, "Hierarchical Web caching systems: modeling, design and experimental results," *Selected Areas in Communications, IEEE Journal on*, vol. 20, no. 7, pp. 1305–1314, Sep 2002.

[2] "Cisco visual networking index: Global mobile data traffic forecast update, 20162021 white paper," CISCO, Tech. Rep., February 2017.

[3] K. Shanmugam *et al.*, "Femtocaching: Wireless video content delivery through distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[4] N. Golrezaei *et al.*, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *IEEE INFOCOM 2012*, March 2012, pp. 1107–1115.

[5] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation algorithms for mobile data caching in small cell networks," *IEEE Transactions on Communications*, vol. 62, no. 10, pp. 3665–3677, Oct 2014.

[6] K. Naveen *et al.*, "On the interaction between content caching and request assignment in cellular cache networks," in *5th Workshop on All Things Cellular: Oper., Applic, and Challenges*. ACM, 2015.

[7] A. Tuholukova, G. Neglia, and T. Spyropoulos, "Optimal cache allocation for Femto helpers with joint transmission capabilities," in *ICC 2017, IEEE International Conference on Communications, Communications QoS, Reliability, and Modeling Symposium, 21-25 May 2017, Paris, France*, Paris, FRANCE, 05 2017. [Online]. Available: http://www.eurecom.fr/publication/5219

[8] A. Chattopadhyay and B. Blaszczyszyn, "Gibbsian on-line distributed content caching strategy for cellular networks," *IEEE Trans. on Wireless Communications*, vol. 17, no. 2, 2018.

[9] B. Blaszczyszyn and A. Giovanidis, "Optimal geographic caching in cellular networks," in *IEEE ICC 2015*, June 2015, pp. 3358–3363.

[10] K. Avrachenkov, J. Goseling, and B. Serbetci, "A low-complexity approach to distributed cooperative caching with geographic constraints," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 1, pp. 27:1–27:25, Jun. 2017.

[11] M. Leconte *et al.*, "Placing dynamic content in caches with small population," in *IEEE INFOCOM 2016*, 2016.

[12] A. Giovanidis and A. Avranas, "Spatial multi-lru caching for wireless networks with coverage overlaps," 2016, arXiv:1612.04363.

[13] S.-E. Elayoubi and J. Roberts, "Performance and cost effectiveness of caching in mobile access networks," in *Proceedings of the 2Nd ACM Conference on Information-Centric Networking*, ser. ACM-ICN '15. New York, NY, USA: ACM, 2015, pp. 79–88.

[14] M. Garetto, E. Leonardi, and V. Martina, "A unified approach to the performance analysis of caching systems," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 1, no. 3, pp. 12:1–12:28, May 2016.

[15] S. Traverso *et al.*, "Temporal Locality in Today's Content Caching: Why It Matters and How to Model It," *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 5, pp. 5–12, Nov. 2013.

[16] G. Neglia, D. Carra, M. Feng, V. Janardhan, P. Michiardi, and D. Tsigkari, "Access-time-aware cache algorithms," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 2, no. 4, pp. 21:1–21:29, Nov. 2017.

[17] A. Dan and D. Towsley, "An approximate analysis of the lru and fifo buffer replacement schemes," in *Proceedings of the 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '90. New York, NY, USA: ACM, 1990, pp. 143–152.

[18] R. Fagin, "Asymptotic miss ratios over independent references," *Journal of Computer and System Sciences*, vol. 14, no. 2, pp. 222 – 250, 1977.

[19] M. Garetto, E. Leonardi, and S. Traverso, "Efficient analysis of caching strategies under dynamic content popularity," in *IEEE INFOCOM 2015*, April 2015, pp. 2263–2271.

[20] P. R. Jelenkovic, "Asymptotic approximation of the move-to-front search cost distribution and least-recently used caching

fault probabilities," *The Annals of Applied Probability*, vol. 9, no. 2, pp. 430–464, 1999.
[21] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for LRU cache performance," in *Proceedings of the 24th International Teletraffic Congress*, 2012, p. 8.
[22] N. C. Fofack, P. Nain, G. Neglia, and D. Towsley, "Performance evaluation of hierarchical TTL-based cache networks," *Computer Networks*, vol. 65, pp. 212 – 231, 2014.
[23] R. W. Wolff, *Stochastic modeling and the theory of queues*. Pearson College Division, 1989.
[24] A. Pelizzola, "Cluster variation method in statistical physics and probabilistic graphical models," *Journal of Physics A: Mathematical and General*, vol. 38, no. 33, p. R309, 2005.
[25] G. Neglia *et al.*, "Access-time aware cache algorithms," in *ITC-28*, September 2016.
[26] H. P. Young, "The Evolution of Conventions," *Econometrica*, vol. 61, no. 1, pp. 57–84, January 1993.
[27] "Openmobilenetwork." [Online]. Available: http://map.openmobilenetwork.org/
[28] G. Neglia, D. Carra, and P. Michiardi, "Cache Policies for Linear Utility Maximization," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 302–313, 2018. [Online]. Available: https://doi.org/10.1109/TNET.2017.2783623
[29] P. Konstantopoulos and J. Walrand, "A quasi-reversibility approach to the insensitivity of generalized semi-markov processes," *Probability in the Engineering and Informational Sciences*, vol. 3, no. 3, pp. 405–415, 1989.

APPENDIX I
INSENSITIVITY FOR FIFO CACHES

The proof of Proposition III.1 follows.

Proof. The vector $\mathbf{X}_f(t)$ indicates where copies of content f are stored at time t. Under CTA for FIFO, when a copy is inserted at cache b, a timer is set to the deterministic value $T_c^{(b)}$ and decreased over time. When the timer reaches zero, the content is erased from cache b. We denote by $c_f^{(b)}(t)$ the residual value of such timer at time t. The system state at time t, as regards content f, is then characterized by $\mathbf{Y}_f(t) \triangleq (\mathbf{X}_f(t), \{c_f^{(b)}(t), \forall b \ | X_f^{(b)}(t) = 1\})$, i.e. by the current allocation of content f copies and their residual timers. The process $\mathbf{Y}_f(t)$ is a Generalized Semi-Markov Process (GSMP) [29].

Let $\pi(\mathbf{x}_f)$ be the stationary distribution of $\mathbf{X}_f(t)$. This distribution is in general a function of the timer distributions. If it depends on them only through their expected values, then the GSMP is said to be *insensitive*. In this case the stationary distribution remains unchanged if we replace all the timers with exponential random variables and then the GSMP simply becomes a continuous time Markov Chain with state $\mathbf{X}_f(t)$. Then, Approximation A1 is correct whenever the GSMP $\mathbf{Y}_f(t)$ is insensitive.

A GSMP is insensitive if and only if its stationary distribution $\pi(\mathbf{x}_f)$ satisfies some partial balance equations (a.k.a. Matthes' conditions) [29, Eq. (2.2)], as well as the usual global balance equations. For our system, Matthes' conditions can be written as

$$\pi\left(0, \mathbf{x}_f^{(-b)}\right) \Lambda_f^{(b)} = \pi\left(1, \mathbf{x}_f^{(-b)}\right) \frac{1}{T_c^{(b)}} \ \ \forall \mathbf{x}_f^{(-b)}, \quad (6)$$

i.e. the rate at which the timer c_f^b is activated is equal to the rate at which it expires. Conditions (6) are equivalent to the corresponding MC being reversible. This completes the proof.

□

APPENDIX II
QLRU-LAZY'S OPTIMALITY

In the trefoil topology, any cell is equivalent to any other, so it does not matter where the copies of a given content are located, but just how many of them there are. Let $\Lambda_f(k)$ be the total request rate for content f from users located in k cells,[12] i.e.:

$$\Lambda_f(k) \triangleq \lambda_f \mu \left(\bigcup_{b=1}^{k} S_b \right).$$

$\Lambda_f(k)$ corresponds then to content f hit ratio when k copies of the content are stored at the caches. Moreover, we denote by $\Delta\Lambda_f(k) \triangleq \Lambda_f(k) - \Lambda_f(k-1)$ the marginal increase of the hit ratio due to adding a k-th copy of content f. We observe that $\Delta\Lambda_f(k)$ is decreasing in k and that it holds:

$$\Lambda_f(k) = \sum_{h=1}^{k} \Delta\Lambda_f(h).$$

A. Proof of Proposition V.1

Proof. The proof is rather immediate. First observe that by exploiting the properties of the cell-trefoil topology, (4) can be rewritten as:

$$\begin{aligned} &\text{maximize} \sum_{f=1}^{F} \Lambda_f(k_f), \quad (7) \\ &\text{subject to} \sum_{f=1}^{F} k_f = B \times C. \end{aligned}$$

Now, caching problem (7) can be easily mapped to a (trivial) knapsack problem with $F \times C$ objects of unitary size, according to the following lines: for every content f we define C different virtual objects (f, h) with $1 \leq h \leq C$, with associated weights:

$$w_{(f,h)} = \Delta\Lambda_f(h),$$

[12] It does not matter which ones, because of the symmetry.

i.e. the weight $w_{(f,h)}$ is equal to the marginal increase of the hit ratio, which is obtained by storing the h-th copy of content f into the caching system.

The objective of the knapsack problem is to find the set $\mathcal{S}_{\text{opt}}$ of $F \times C$ objects, which maximizes the sum of all the associated weights. Indeed (7) can be rewritten as: $\max_{(k_1,\dots,k_f,\dots,k_F)} \sum_f \sum_{h=1}^{k_f} w_{(f,h)}$. In particular, observe that since $w_{(f,h)} \leq w_{(f,h-1)}$, virtual object $(f,h) \in \mathcal{S}_{\text{opt}}$ only if $(f,h-1) \in \mathcal{S}_{\text{opt}}$. This implies that $\mathcal{S}_{\text{opt}}$ provides a feasible solution for the original caching problem, where k_f is equal to the largest h such that $(f,h) \in \mathcal{S}_{\text{opt}}$.

Finally, note that, by construction, $\mathcal{S}_{\text{opt}}$ is the set composed of the $F \times C$ objects with the largest value; therefore by construction, $\mathcal{S}_{\text{opt}}$ corresponds to: i) the caching allocation that maximizes the global hit rate (i.e. the allocation that solves (7)); ii) moreover, it is the only solution of the greedy algorithm, under the assumption that object values are all different, (i.e. for generic values of the parameters).

□

B. Proof of Proposition V.2

Proof. Under our approximated model, system dynamics are described by F Markov chains, one for each content, coupled by the characteristic times. The symmetry of the trefoil topology implies that the characteristic time at each cache has the same value that we denote simply as T_c.

Every MC is a birth-death process. In particular, under QLRU-LAZY, for content f, the transition rate from state $k_f - 1$ to k_f is

$$r(k_f - 1, k_f) \triangleq q\left(\Lambda_f(B) - \Lambda_f(k_f - 1)\right),$$

and from state k_f to $k_f - 1$ it is

$$r(k_f, k_f - 1) \triangleq k_f \frac{\Delta\Lambda_f(k_f)}{e^{\Delta\Lambda_f(k_f)T_c} - 1}.$$

Let us define $\rho_f(k) \triangleq r(k_f - 1, k_f)/r(k_f, k_f - 1)$. The stationary probability to have k_f copies of content f is then

$$\begin{aligned}\pi_f(k_f) &= \frac{\prod_{h=1}^{k_f} \rho_f(h)}{1 + \sum_{k=1}^{B} \prod_{h=1}^{k} \rho_f(h)} \\ &= \frac{A_{f,k_f} \prod_{h=1}^{k_f} q\left(e^{\Delta\Lambda_f(h)T_c} - 1\right)}{1 + \sum_{k=1}^{B} A_{f,k} \prod_{h=1}^{k} q\left(e^{\Delta\Lambda_f(h)T_c} - 1\right)},\end{aligned}$$

where

$$A_{f,k} \triangleq \prod_{h=1}^{k} \frac{\Lambda_f(B) - \Lambda_f(h-1)}{h\Delta\Lambda_f(h)}$$

are values that do not depend on q or T_c and they will not play a role in the following study of the asymptotic behaviour.

Under CTA, the buffer constraint is expressed imposing that the expected number of contents at a cache is equal to the buffer size. If the system is in state k_f, any given BS has probability k_f/B to be one of the k_f storing it, then the buffer constraint is

$$\sum_{f=1}^{F} \sum_{k=1}^{B} \frac{k}{B} \pi_f(k) = C,$$

or equivalently:

$$\sum_{f=1}^{F} \sum_{k=1}^{B} k\pi_f(k) = C \times B. \tag{8}$$

We focus now our attention on the stationary distribution when q converges to 0. As q changes, the characteristic time changes as well. We write $T_c(q)$ to express such dependence. When q converges to 0, $T_c(q)$ diverges, otherwise all the probabilities $\pi_f(k_f)$ would converge to 0 and constraint (8) would not be satisfied. It follows that:

$$\pi_f(q, k_f) \underset{q\to 0}{\sim} \frac{A_{f,k_f} \prod_{h=1}^{k_f} \left(qe^{\Delta\Lambda_f(h)T_c(q)}\right)}{1 + \sum_{k=1}^{B} A_{f,k} \prod_{h=1}^{k} \left(qe^{\Delta\Lambda_f(h)T_c(q)}\right)}. \tag{9}$$

Let us consider a sequence $(q_n)_{n\in\mathbb{N}}$ that converges to 0 such that it exists $\lim_{n\to\infty} \ln(1/q_n)/T_c(q_n) = \hat{\Lambda}$. The value $\hat{\Lambda}$ is also said to be a cluster value for the function $\ln(1/q)/T_c(q)$. It holds that for any marginal hit ratio $\Delta\Lambda_f(h)$

$$\lim_{n\to\infty} q_n e^{\Delta\Lambda_f(h)T_c(q_n)} = \begin{cases} +\infty, & \text{if } \Delta\Lambda_f(h) > \hat{\Lambda}, \\ 0, & \text{if } \Delta\Lambda_f(h) < \hat{\Lambda}. \end{cases}$$

Let $\hat{k}_f \triangleq \arg\max_h\{\Delta\Lambda_f(h) > \hat{\Lambda}\}$. For generic values of the parameters the values $\{\Delta\Lambda_f(h), \forall f = 1,\dots F, h = 1,\dots B\}$ are all distinct by hypothesis, then there can be at most one content f_0 for which it holds $\Delta\Lambda_f(\hat{k}_{f_0} + 1) = \hat{\Lambda}$. For any other content it follows that the dominant term in the denominator of (9) is $\prod_{h=1}^{\hat{k}_f} \left(qe^{\Delta\Lambda_f(h)T_c(q)}\right)$ and then for $f \neq f_0$:

$$\lim_{n\to\infty} \pi_f(q_n, k_f) = \begin{cases} 1, & \text{if } k_f = \hat{k}_f, \\ 0, & \text{otherwise}, \end{cases}$$

i.e. asymptotically exactly $\hat{k}_f$ copies of content f would be stored. For content f_0, both the term $\prod_{h=1}^{\hat{k}_{f_0}} \left(qe^{\Delta\Lambda_f(h)T_c(q)}\right)$ and $\prod_{h=1}^{\hat{k}_{f_0}+1} \left(qe^{\Delta\Lambda_f(h)T_c(q)}\right)$ could be dominant. Then all the $\pi_{f_0}(q_n, k_f)$ converge to 0 for

$k_f \notin \{\hat{k}_{f_0}, \hat{k}_{f_0}+1\}$. The total expected number of copies stored in the system would then be:

$$\sum_{f\neq f_0} \hat{k}_f + \hat{k}_{f_0}\pi_{f_0}(0,\hat{k}_{f_0}) + (\hat{k}_{f_0}+1)\pi_{f_0}(0,\hat{k}_{f_0}+1) = \sum_{f\neq f_0} \hat{k}_f + \hat{k}_{f_0} + \pi_{f_0}(0,\hat{k}_{f_0}+1).$$

Because of (8), this sum has to be equal to the integer $C\times B$, then one of the two following mutually exclusive possibilities must hold, or

$$\sum_{f\neq f_0} \hat{k}_f + \hat{k}_{f_0} = C\times B$$

and then $\pi_{f_0}(0,\hat{k}_{f_0}+1)=0$, or

$$\sum_{f\neq f_0} \hat{k}_f + \hat{k}_{f_0} + 1 = C\times B$$

and then $\pi_{f_0}(0,\hat{k}_{f_0}+1)=1$. In any case, the conclusion is that, when q converges to 0, for each content f a fixed number of copies k_f is stored at the cache. k_f is such that the marginal hit-ratio increase due to the k_f-th copy is among the largest $C\times B$ marginal hit-ratios (and the (k_f+1)-th copy is not among them). This allocation coincides with the solution of the greedy algorithm. □

C. Proof of Proposition V.3

Proof. For a given content f, let $\mathbf{x}_f$ and $\mathbf{y}_f$ be two possible states of the MC. We say that $\mathbf{x}_f \leq \mathbf{y}_f$ whenever $x_f^{(b)} \leq y_f^{(b)}$ for each b; furthermore we denote with $|\mathbf{x}_f| = \sum_b x_f^{(b)}$ the number of stored copies of the content in state $\mathbf{x}_f$, which we call weight of the state $\mathbf{x}_f$.

Now observe that by construction, transition rates in the MC are different from 0 only between pair of states $\mathbf{x}_f$ and $\mathbf{y}_f$, such that: i) $\mathbf{x}_f \leq \mathbf{y}_f$, ii) $|\mathbf{x}_f| = |\mathbf{y}_f| - 1$. In such a case we say that $\mathbf{y}_f$ is a parent of $\mathbf{x}_f$ and $\mathbf{x}_f$ is a son of $\mathbf{y}_f$. Moreover we say that $\mathbf{x}_f \to \mathbf{y}_f$ is an *upward* transition, while $\mathbf{y}_f \to \mathbf{x}_f$ is a downward transition.

Let $\mathbf{y}_f$ be parent of $\mathbf{x}_f$ and let b_0 be the index such that $x_f^{(b_0)} < y_f^{(b_0)}$, we have that the upward rate $\rho_{[\mathbf{x}_f\to\mathbf{y}_f]} = q\Lambda_f^{(b_0)}(\mathbf{x}_f) = \Theta(q)$ and the downward rate $\rho_{[\mathbf{y}_f\to\mathbf{x}_f]} = \frac{\Lambda_f^{(b_0)}(\mathbf{x}_f)}{e^{\Lambda_f^{(b_0)}(\mathbf{x}_f)T_C^{(b_0)}}-1}$.

Now, as $q\to 0$ for every f every upward rate $r_{[\mathbf{x}_f\to\mathbf{y}_f]}$ tends to 0. Therefore necessarily the characteristic time of every cell $T_C^{(b)}$ must diverges. In fact, if it were not the case for a cache b, none of the contents would be found in the cache b asymptotically, because upward rates tend to zero, while downward rates would not. This would contradict the constraint:

$$\sum_f \sum_{\mathbf{x}_f} x_f^{(b)}\pi(\mathbf{x}_f) = C \qquad \forall b \tag{10}$$

imposed by the CTA. Therefore necessarily $T_C^{(b)}\to\infty$ for every cell b. More precisely we must have $T_C^{(b)} = \Theta(\log\frac{1}{q})$ at every cache otherwise we fail to meet (10). Now we can always select a sequence $\{q_n\}_n$ such that $T_C^{(b)}(q_n) \sim \frac{1}{\gamma_b}(\log\frac{1}{q_n})$.

Let us now consider the uniformization of the continuous time MC $\mathbf{X}_f(t)$ with an arbitrarily high rate Λ_T and the corresponding discrete time MC $\mathbf{X}_f(k)$ with transition probability matrix $P_{f,q}$. For $q=0$, the set of contents in the cache does not change, each state is an absorbing one and any probability distribution is a stationary probability distribution for $P_{f,0}$. We are rather interested in the asymptotic behaviour of the MC when q converges to 0. For $q>0$ the MC is finite, irreducible and aperiodic and then admits a unique stationary probability $\pi_{f,q}$. We call the states $\mathbf{x}_f$ for which $\lim_{q\to 0}\pi_{f,q}(\mathbf{x}_f) > 0$ *stochastically stable*. We are going to characterize such states.

For what we have said above, it holds that the probability to move from $\mathbf{x}$ to the parent $\mathbf{y}$ is $P_{f,q}(\mathbf{x}_f,\mathbf{y}_f) \sim \Lambda_f^{(b_0)}q$, while $P_{f,q}(\mathbf{y}_f,\mathbf{x}_f) \sim \Lambda_f^{(b_0)}q^{\Lambda_f^{(b_0)}(\mathbf{x}_f)/\gamma_{b_0}}$. For each possible transition, we define its *direct resistance* to be the exponent of the parameter q, then $r(\mathbf{x}_f,\mathbf{y}_f)=1$, $r(\mathbf{y}_f,\mathbf{x}_f) = \Lambda_f^{(b_0)}(\mathbf{x}_f)/\gamma_{b_0}$ and $r(\mathbf{x}_f,\mathbf{x}_f)=0$. If a direct transition is not possible between two states, then we consider the corresponding direct resistance to be infinite. Observe that the higher the resistance, the less likely the corresponding transition. Given a sequence of transitions $(\mathbf{x}_f^1,\mathbf{x}_f^2\ldots\mathbf{x}_f^n)$ from state $\mathbf{x}_f^1$ to state $\mathbf{x}_f^n$, we define its resistance to be the sum of the resistances, i.e. $r(\mathbf{x}_f^1,\mathbf{x}_f^2\ldots\mathbf{x}_f^n) = \sum_{i=1}^{n-1} r(\mathbf{x}_f^i,\mathbf{x}_f^{i+1})$.

The family of Markov chains $\{P_{f,q}\}$ is a *regular perturbation* [26, properties (6-8)] and then it is possible to characterize the stochastically stable states as the minimizers of the potential function $V_f(\mathbf{x}_f)$ defined as follows. For each pair of states $\mathbf{x}_f$ and $\mathbf{x}'_f$ let $R(\mathbf{x}_f,\mathbf{x}'_f)$ be the minimum resistance of all the possible sequences of transitions from $\mathbf{x}_f$ to $\mathbf{x}'_f$ (then $R(\mathbf{x}_f,\mathbf{x}'_f)\leq r(\mathbf{x}_f,\mathbf{x}'_f)$). Consider then the full meshed directed weighted graph whose nodes are the possible states of the MC and the weights of the edge $(\mathbf{x}_f,\mathbf{x}'_f)$ is $R(\mathbf{x}_f,\mathbf{x}'_f)$. The potential of state $\mathbf{x}_f$ ($V_f(\mathbf{x}_f)$) is defined as the resistance of the minimum weight in-tree (or anti-arborescence) rooted to $\mathbf{x}_f$. Intuitively the potential is a measure of the general difficulty to reach state $\mathbf{x}_f$ from all the other nodes. From

Theorem 4 of [26] it follows that $\mathbf{x}_f$ is stochastically stable if and only if its potential is minimal.

For each content f we are then able to characterize which configurations are stochastically stable as q converges to 0. Moreover, this set of configurations must satisfy the constraint (10) at each base station b. We define then the cache configuration $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_F)$ to be *jointly stochastically stable* if 1) for each content f $\mathbf{x}_f$ is stochastically stable, 2) $\mathbf{x}$ satisfies (10) for each b.

The last step in order to prove Proposition (V.3) is to show that a jointly stochastically stable cache configuration $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_F)$ is locally optimal, i.e. that changing one content at a given cache does not increase the hit ratio. Without loss of generality, we consider to replace content f_1 present at cache B with content f_2. Then, the cache allocation $\mathbf{x}$ changes from $\mathbf{x}_{f_1} = (x_{f_1}^{(B)} = 1, \mathbf{x}_{f_1}^{(-B)})$ and $\mathbf{x}_{f_2} = (x_{f_2}^{(B)} = 0, \mathbf{x}_{f_2}^{(-B)})$ to a new one cache allocation $\mathbf{x}'$, such that $\mathbf{x}'_{f_1} = (x_{f_1}^{\prime(B)} = 0, \mathbf{x}_{f_1}^{(-B)})$ and $\mathbf{x}'_{f_2} = (x_{f_2}^{\prime(B)} = 1, \mathbf{x}_{f_2}^{(-B)})$. Let $\eta_f(\mathbf{x}_f)$ denote the hit rate for content f over the whole network under the allocation $\mathbf{x}_f$ and $\eta(\mathbf{x} = (\mathbf{x}_1, \dots \mathbf{x}_F)) = \sum_{f=1}^{F} \eta_f(\mathbf{x}_f)$ the global hit rate across all the contents. Lemma II.1 below provides a formula for the hit rate $\eta(\mathbf{x})$, from which we obtain that

$$\begin{aligned}\eta(\mathbf{x}) \geq \eta(\mathbf{x}') &\Leftrightarrow \eta_{f_1}(\mathbf{x}_{f_1}) + \eta_{f_2}(\mathbf{x}_{f_2}) \\ &\quad \geq \eta_{f_1}(\mathbf{x}'_{f_1}) + \eta_{f_2}(\mathbf{x}'_{f_2}) \\ &\Leftrightarrow \Lambda_{f_1}^{(B)}(\mathbf{x}_{f_1}^{(-B)}, 0) \geq \Lambda_{f_2}^{(B)}(\mathbf{x}_{f_2}^{(-B)}, 0) \\ &\Leftrightarrow \Lambda_{f_1}^{(B)}(\mathbf{x}'_{f_1}) \geq \Lambda_{f_2}^{(B)}(\mathbf{x}_{f_2}). \end{aligned} \tag{11}$$

In order to prove (11), we will show that

$$\Lambda_{f_1}^{(B)}(\mathbf{x}'_{f_1}) \geq \gamma_B \tag{12}$$

$$\Lambda_{f_2}^{(B)}(\mathbf{x}_{f_2}) \leq \gamma_B. \tag{13}$$

The state $\mathbf{x}'_{f_1}$ is a child of $\mathbf{x}_{f_1}$, then $r(\mathbf{x}_{f_1}, \mathbf{x}'_{f_1}) = \Lambda_{f_1}^{(B)}(\mathbf{x}'_{f_1})/\gamma_B$. Consider the in-tree $\mathcal{T}$ rooted in $\mathbf{x}_{f_1}$ with minimal resistance and let $R(\mathcal{T})(= V(\mathbf{x}_{f_1}))$ denote its resistance and $(\mathbf{x}_{f_1}^1 = \mathbf{x}'_{f_1}, \mathbf{x}_{f_1}^2, \dots \mathbf{x}_{f_1}^k = \mathbf{x}_{f_1})$ be the sequence of transitions in $\mathcal{T}$ from $\mathbf{x}'_{f_1}$ to $\mathbf{x}_{f_1}$. One of these transitions, say it $\mathbf{x}_{f_1}^l \to \mathbf{x}_{f_1}^{l+1}$ corresponds to store the content f_1 in the cache B and has resistance 1. Consider now the in-tree $\mathcal{T}'$ rooted in $\mathbf{x}_{f_1}^l$ obtained from $\mathcal{T}$ removing the edge $(\mathbf{x}_{f_1}^l, \mathbf{x}_{f_1}^{l+1})$ and adding the edge $(\mathbf{x}_{f_1}, \mathbf{x}'_{f_1})$. Its resistance is $R(\mathcal{T}') = R(\mathcal{T}) - 1 + \Lambda_{f_1}^{(B)}(\mathbf{x}'_{f_1})/\gamma_B$. From $R(\mathcal{T}') \geq V(\mathbf{x}_{f_1}^l) \geq V(\mathbf{x}_{f_1}) = R(\mathcal{T})$ it follows (12). A sketch of this construction is in Fig. 10.

The proof of (13) is slightly more complex. It is useful to introduce some additional definitions. Given two neighboring states $\mathbf{x}_f$ and $\mathbf{x}'_f$, we say that the transition $\mathbf{x}_f \to \mathbf{x}'_f$ is *dominant* if $r(\mathbf{x}_f, \mathbf{x}'_f) \leq r(\mathbf{x}'_f, \mathbf{x}_f)$. Let $\mathbf{y}_f$ be a parent of $\mathbf{x}_f$ with $x_f^{(b)} = 0$ and $y_f^{(b)} = 1$, we observe that the upward transition $\mathbf{x}_f \to \mathbf{y}_f$ is dominant if and only if $\Lambda_f^{(b)}(\mathbf{x}_f) \geq \gamma_b$. Similarly the downward transition $\mathbf{y}_f \to \mathbf{x}_f$ is dominant if and only if $\Lambda_f^{(b)}(\mathbf{x}_f) \leq \gamma_b$. Let us also consider the function of state $\mathbf{x}_f$

$$\phi(\mathbf{x}_f) \triangleq \lambda_f \mu \left(\bigcup_{b \mid x_f^{(b)}=1} S_b \right) - \sum_{b \mid x_f^{(b)}=1} \gamma_b. \tag{14}$$

Lemma II.2 guarantees that the function $\phi(.)$ cannot decrease along a dominant transition.

First we prove our result under the assumption that $\gamma_b = \gamma$ for every cell b, then we provide the generalization to the most general case. Let us prove (13) by contradiction assuming that $\Lambda_{f_2}^{(B)}(\mathbf{x}_{f_2}) > \gamma_B$. In such case $\mathbf{x}'_{f_2} \to \mathbf{x}_{f_2}$ is not a dominant (downward) transition, and $\phi(\mathbf{x}_{f_2}) < \phi(\mathbf{x}'_{f_2})$.

Let now $\mathcal{T}$ denote the in-tree rooted in $\mathbf{x}_{f_2}$ with minimal resistance and $\mathcal{P} = (\mathbf{x}_{f_2}^1 = \mathbf{x}'_{f_2}, \mathbf{x}_{f_2}^2, \dots \mathbf{x}_{f_2}^k = \mathbf{x}_{f_2})$ be the sequence of transitions in $\mathcal{T}$ from $\mathbf{x}'_{f_2}$ to $\mathbf{x}_{f_2}$. At each transition $\mathbf{x}_{f_2}^l \to \mathbf{x}_{f_2}^{l+1}$ only one state variable changes, we denote by b_l the corresponding index, representing the base station at/from which a copy of content f_2 is added/removed. By construction we have:

$$0 > \phi(\mathbf{x}_{f_2}) - \phi(\mathbf{x}'_{f_2}) = \sum_{1 \leq l \leq k-1} \phi(\mathbf{x}_{f_2}^{l+1}) - \phi(\mathbf{x}_{f_2}^l)$$

Now observe that:

$$\begin{aligned}&\phi(\mathbf{x}_{f_2}^{l+1}) - \phi(\mathbf{x}_{f_2}^l) \\ &\quad = \Lambda_{f_2}^{(b_l)}(\mathbf{x}_{f_2}^l) - \gamma_{b_l} = \Lambda_{f_2}^{(b_l)}(\mathbf{x}_{f_2}^{l+1}) - \gamma_{b_l} \\ &\quad = \gamma_{b_l}[r(\mathbf{x}_{f_2}^l, \mathbf{x}_{f_2}^{l+1}) - r(\mathbf{x}_{f_2}^{l+1}, \mathbf{x}_{f_2}^l)] \end{aligned} \tag{15}$$

if transition $\mathbf{x}_{f_2}^l \to \mathbf{x}_{f_2}^{l+1}$ is upward, and

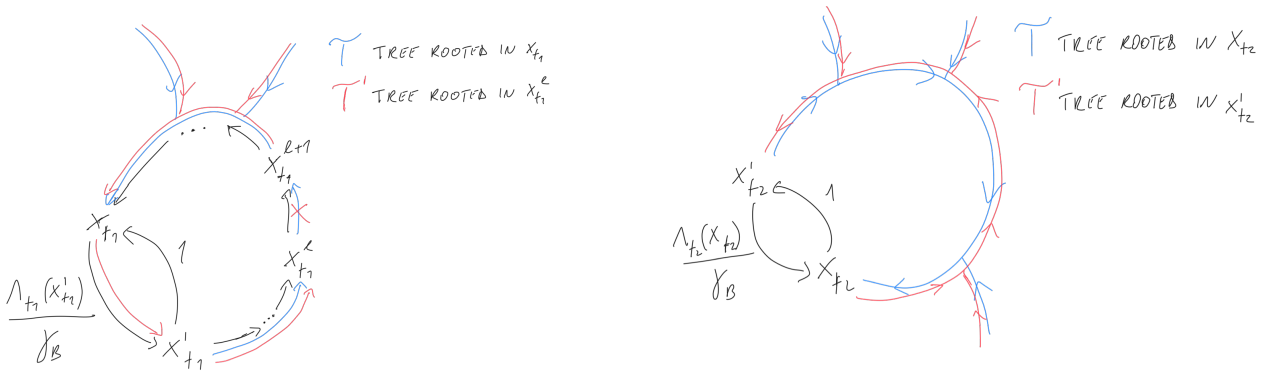
$$\begin{aligned}\phi(\mathbf{x}_{f_2}^{l+1}) - \phi(\mathbf{x}_{f_2}^l) &= \gamma_{b_l} - \Lambda_{f_2}^{(b_l)}(\mathbf{x}_{f_2}^l) \\ &= \gamma_{b_l} - \Lambda_{f_2}^{(b_l)}(\mathbf{x}_{f_2}^{l+1}) \\ &= \gamma_{b_l}[r(\mathbf{x}_{f_2}^l, \mathbf{x}_{f_2}^{l+1}) - r(\mathbf{x}_{f_2}^{l+1}, \mathbf{x}_{f_2}^l)] \end{aligned} \tag{16}$$

if transition $\mathbf{x}_{f_2}^l \to \mathbf{x}_{f_2}^{l+1}$ is downward. Therefore

$$\begin{aligned}&\phi(\mathbf{x}_{f_2}) - \phi(\mathbf{x}'_{f_2}) = \\ &\quad \sum_{1 \leq l \leq k-1} \gamma_{b_l}[r(\mathbf{x}_{f_2}^l, \mathbf{x}_{f_2}^{l+1}) - r(\mathbf{x}_{f_2}^{l+1}, \mathbf{x}_{f_2}^l)] < 0 \end{aligned} \tag{17}$$

From which, under the assumption $\gamma_b = \gamma$ for any b, we have:

$$\sum_{1 \leq l \leq k-1} r(\mathbf{x}_{f_2}^l, \mathbf{x}_{f_2}^{l+1}) > \sum_{1 \leq l \leq k-1} r(\mathbf{x}_{f_2}^{l+1}, \mathbf{x}_{f_2}^l) \tag{18}$$



(a) Proof of Eq. (12)

(b) Proof of Eq. (13)

Fig. 10. Sketch of the constructions used to prove Proposition V.3.

where the term on the LHS is the total resistance of path $\mathcal{P}$ while the term on the RHS is the total resistance of the reverse path $\widehat{\mathcal{P}} = (\widehat{\mathbf{x}}^1_{f_2} = \mathbf{x}_{f_2}, \widehat{\mathbf{x}}^2_{f_2} = \widehat{\mathbf{x}}^{k-1}_{f_2}, \dots \widehat{\mathbf{x}}^k_{f_2} = \mathbf{x}'_{f_2})$.

Hence if we consider the in-tree $\mathcal{T}'$ routed at $\mathbf{x}'_{f_2}$, which is obtained from $\mathcal{T}$ by reverting all the edges of $\mathcal{P}$, i.e. $\mathcal{T}' = \mathcal{T} - \mathcal{P} + \widehat{\mathcal{P}}$, we obtain that $r(\mathcal{T}') < r(\mathcal{T})$ contradicting the hypothesis. A sketch of this construction is in Fig. **??**.

In the most general case, i.e. when γ_b are different, a set of QLRU-LAZY caches all with the same parameter q is not anymore guaranteed to be locally optimal (previous proof fails because from (17) we cannot deduce (18)). However we can still define a provable locally optimal scheme, if we allow the adoption of different parameters q at different cells for the implementation of the local QLRU policy. In particular by selecting $q_b = (q_n)^{\gamma_b}$ we can force characteristic times $T_C^{(b)}$ to be asymptotically equal at different cells.

Direct resistances for our generalized scheme satisfy: $r(\mathbf{x}_f, \mathbf{y}_f) = \gamma_b$, $r(\mathbf{y}_f, \mathbf{x}_f) = \Lambda_f^{(b_0)}(\mathbf{x}_f)$ when $\mathbf{y}$ is chosen to be a parent of $\mathbf{x}$. As a consequence, in this case, we have:

$$\phi(\mathbf{x}_f) - \phi(\mathbf{y}_f) = r(\mathbf{x}_f, \mathbf{y}_f) - r(\mathbf{x}_f, \mathbf{y}_f)$$

and

$$\phi(\mathbf{y}_f) - \phi(\mathbf{x}_f) = r(\mathbf{y}_f, \mathbf{x}_f) - r(\mathbf{y}_f, \mathbf{x}_f)$$

Hence by repeating exactly the same arguments as for the special case $\gamma_b = \gamma$, our generalized scheme can proved to be locally optimal.

□

Lemma II.1. *Given a cache configuration* $\mathbf{x} = (\mathbf{x}_1, \dots \mathbf{x}_F)$*, under* QLRU-LAZY *the hit rate for content* f *can be calculated as follows*

$$\eta_f(\mathbf{x}_f) \triangleq \sum_{b=1}^{B} \mathbb{1}(x_f^{(b)} = 1) \Lambda_f^{(b)}(x_f^{(1)}, \dots x_f^{(b-1)}, 0, \dots, 0),$$

and the global hit rate is

$$\eta(\mathbf{x}) \triangleq \sum_{f=1}^{F} \eta_f(\mathbf{x}_f).$$

Proof. The hit rate for content f is

$$\begin{aligned}
\eta_f(\mathbf{x}_f) &= \lambda_f \mu \left(\bigcup_{b | x_{f_2}^{(b)} = 1} S_b \right) \\
&= \lambda_f \sum_{b=1}^{B} \mathbb{1}(x_f^{(b)} = 1) \mu \left(S_b \setminus \bigcup_{b' < b | x_{f_2}^{(b')} = 1} S_b \right) \\
&= \sum_{b=1}^{B} \mathbb{1}(x_f^{(b)} = 1) \Lambda_f^{(b)}(x_f^{(1)}, \dots x_f^{(b-1)}, 0, \dots, 0).
\end{aligned}$$

□

Lemma II.2. *Given a dominant transition* $\mathbf{x}_f \to \mathbf{y}_f$*, it holds* $\phi(\mathbf{y}) \geq \phi(\mathbf{x})$.

Proof. Let b' be the index at which $\mathbf{x}_f$ and $\mathbf{y}_f$ differ. If $\mathbf{x}_f \to \mathbf{y}_f$ is an upward dominant transition, then

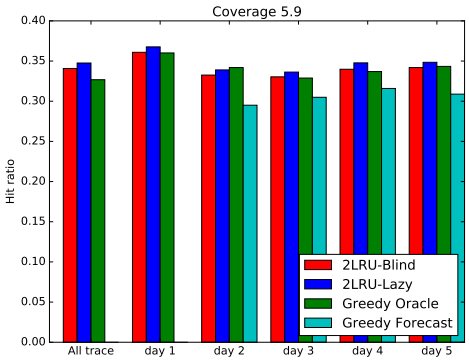


Fig. 11. Berlin topology with average coverage 5.9 and real CDN request trace. Comparison of the different policies over the whole trace and for each of the 5 days.

$\Lambda_f^{(b)}(\mathbf{x}_f) \geq \gamma_b$ and it follows:

$$\begin{aligned}\phi(\mathbf{y}_f) &= \lambda_f \mu \left(\bigcup_{b \mid y_f^{(b)}=1} S_b \right) - \sum_{b \mid y_f^{(b)}=1} \gamma_b \\ &= \lambda_f \mu \left(\bigcup_{b \mid x_f^{(b)}=1} S_b \right) - \sum_{b \mid x_f^{(b)}=1} \gamma_b + \Lambda_f^{(b')}(\mathbf{x}_f) - \gamma_{b'} \\ &\geq \phi(\mathbf{x}_f).\end{aligned}$$

The proof when $\mathbf{x}_f \rightarrow \mathbf{y}_f$ is a downward dominant transition is similar. □

APPENDIX III
COMPARISON WITH [27]

As we mentioned in Sec. VI, one of the conclusions of [13] is that under the current busy-hour demand (40 Mb/s), an ideal prefetching scheme, based on perfect knowledge of content popularities, performs better than reactive policies. The Akamai request trace we used in Sec. VI corresponds to a busy-hour traffic per cell equal to 800Mbps. We decided then to carry on the same experiments illustrated in Fig. 9 sampling our trace by a factor 20. The results are shown in Fig. 11. The conclusion is that, while 2LRU-LAZY is impaired by the lower traffic rate, it still outperforms the static greedy allocation even if clairvoyant day-ahead estimates are available (at least 4 days out of 5). The figure shows also the results for 2LRU-BLIND (called LRU with prefilter in [13]): it appears to be overall slightly worse than the clairvoyant static allocation, but it still performs better than the more realistic static allocation based on day-ahead forecast.