Soft Cache Hits: Improving Performance through Recommendation and Delivery of Related Content

Pavlos Sermpezis, Theodoros Giannakas, Thrasyvoulos Spyropoulos, and Luigi Vigneri

***Abstract*—Pushing popular content to small cells with local storage ("helper" nodes) has been proposed to cope with the ever-growing data demand. Nevertheless, the collective storage of a few nearby helper nodes may not suffice to achieve a high hit rate in practice. In this paper, we introduce the concept of "soft cache hits" (SCH): An SCH occurs if a user's requested content is not in the local cache, but the user can be (partially) satisfied by a related content that is. In case of a cache miss, an application proxy (e.g., YouTube) running close to the helper node (e.g., at a MEC server) can recommend the most related files that are locally cached. This system could be activated during periods of predicted congestion, or for selected users (e.g., low cost plans), to improve cache hit ratio with limited (and tunable) user QoE performance impact. Beyond introducing a model for soft cache hits, our next contribution is to show that the optimal caching policy should be revisited when SCHs are allowed. In fact, we show that optimal caching with SCH is NP-hard *even for a single cache*. To this end, we formulate the optimal femto-caching problem with SCH in a sufficiently generic setup, and propose efficient algorithms with provable performance. Finally, we use a large range of real datasets to corroborate our proposal.**

***Index Terms*—Mobile Edge Caching; Femto-Caching; Soft Cache Hits; Recommendation Systems; Optimization**

I. Introduction

A. Background and Motivation

Mobile edge caching has been identified as one of the five most disruptive enablers for 5G networks [1], both to reduce content access latency and to alleviate backhaul congestion. However, the number of required storage points in future cellular networks will be orders of magnitude more than in traditional CDNs [2] (e.g., 100s or 1000s of small cells (SCs) corresponding to an area covered by a single CDN server today). As a result, the storage space per local edge cache must be significantly smaller to keep costs reasonable. Even if we considered a small subset of the entire Internet catalogue, e.g., a typical torrent catalogue (1.5 PB) or the Netflix catalogue (3 PB), edge cache hit ratio would still be low even with a relatively skewed popularity distribution and more than 1 TB of local storage [3], [4].

Additional caching gains have been sought by researchers, increasing the "effective" cache size visible to each user. This could be achieved by: (a) *Coverage overlaps*, where each user is in the range of multiple cells, thus having access to the aggregate storage capacity of these cells, as in the femto-caching framework [5], [6]. (b) *Coded caching*, where collocated users overhearing the same broadcast channel may benefit from cached content in other users' caches [7]. (c) *Delayed content access*, where a user might wait up to a TTL for her request, during which time more than one cache (fixed [8] or mobile [9], [10], [11], [12]) can be encountered. While each of these ideas can theoretically increase the cache hit ratio (sometimes significantly), the actual practical gains might not suffice by themselves, e.g., due to high enough cell density required for (a), sub-packetization complexity in (b), and imposed delays in (c).

To get around this seeming impasse, we propose to *move away from trying to satisfy every possible user request, and instead try to satisfy the user*. In an Internet which is becoming increasingly entertainment-oriented, one can make the following observations: (a) a user's content requests are increasingly influenced by various recommendation systems (YouTube, Netflix, Spotify, or even Social Networks) [13]; (b) some related contents (e.g. two recent NBA games, two funny cat clips) might have similar utility for a user; in micro-economic terms, these are often called *substitute goods*; we will use the terms *alternative*, *related*, and *substitute* content inter-changeably.

B. Soft Cache Hits: Idea and Implications

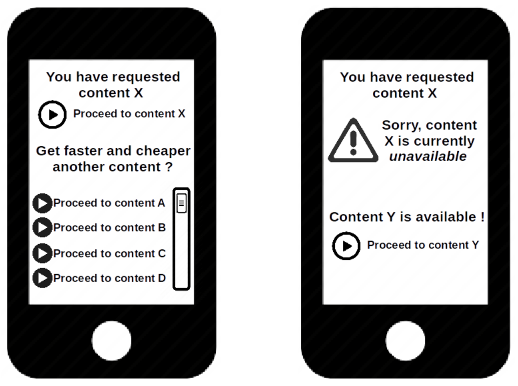
Based on these observations, we envision a system where "soft cache hits" can be leveraged to improve caching performance. As one example, consider the following, for the case of YouTube (or, any similar service). If a user requests a content, e.g., by typing on the YouTube search bar, and the content is not available in the local cache(s), then a local app proxy located near the cache and having knowledge of the cached contents (e.g. a YouTube recommender code running at a Multi-access Edge Computing (MEC) server [14]), could recommend a set of *related contents* that *are* also locally available. If the user prefers or accepts (under some incentives; see below) one of these contents, instead of the one she initially typed/requested, a soft cache hit (SCH) occurs, and an expensive remote access is avoided. We will use the term *soft cache hit* to describe such scenarios.

Of course, appropriate incentives would be needed to nudge a user towards substitute content. While perhaps a somewhat radical concept in today's ecosystem, we believe there are a number of scenarios where soft cache hits are worth considering, as *they could benefit both the user and the operator*. (i) A

Manuscript received December 10, 2017; revised April 8, 2018; accepted April 18, 2018. *(Corresponding author: Pavlos Sermpezis)*

P. Sermpezis is with the Institute of Computer Science, FORTH, 70013 Heraklion, Greece (e-mail: sermpezis@ics.forth.gr).

T. Giannakas, T. Spyropoulos, and L. Vigneri are with the Communications System Department, EURECOM, 06904 Sophia-Antipolis, France (e-mail: first.last@eurecom.fr).



(a) (b)

Fig. 1: Mobile app example for *Soft Cache Hits*: (a) related content *recommendation* (that the user might not accept) , and (b) related content *delivery*.

cache-aware recommendation plugin to an existing application could, for example, let a user know that accessing the original content X is only possible at low quality and might be choppy, freeze, etc., due to congestion, while related contents $A, B, C, ...$ could be streamed at high resolution, as shown in Fig. 1(a). (ii) Alternatively, the operator could activate this system only during predicted congestion periods, while giving some incentives to users to accept the alternative contents during that time (e.g., *zero-rating* services [15], [16]). (iii) In some cases, the operator might even "enforce" an alternative (but related) content (see Fig. 1(b)), e.g., offering low rate plans with higher data quotas with the agreement that, during congestion, only locally cached content can be served.

While, in the above cases, a potential unwillingness or utility loss needs to be counter-balanced with appropriate incentives, this is not always the case. Sometimes soft cache hits could be leveraged in a relatively seamless manner without the potential psychological impact on user quality of experience (QoE) due to *conscient* content replacement[1]. For example, after a user watches a video X, the recommendation system could re-order its list of recommendations (among related contents of roughly equal similarity to X) to favor a cache hit in the next request, without the user being aware of this change or liking the recommended contents less. Such systems have already been considered, and would be complementary to our proposal [18], [19]. A similar case could be made for online radio type of apps (like lastFM, Pandora, Spotify, etc.). While a lot more can be said about each of the above preliminary incentive ideas, and plenty more thinking might be needed to go from these to concrete business cases, we believe these suffice to motivate an investigation of the potential impact of soft cache hits.

While soft cache hits could provide some benefits on top of an existing caching policy, a first key observation is that the optimal caching policy might differ, sometimes radically, when soft cache hits are allowed. As a simple example, consider a single cache with a tiny content catalog with contents A, B, C of popularities 3, 2, 2, respectively (e.g. number of requests per minute). If the cache could fit only a single content, traditional caching will choose to store the most popular content (A), leading to a cache hit ratio of $3/(3+2+2)$, approx. 43%. However, assume we knew that 1 out of 2 users requesting A, would be willing to watch content C instead (e.g. because C is highly related to A, and available locally at HD). Same for users requesting content B. Then, caching content C would satisfy all requests for C (2), half the requests for B ($0.5 \cdot 2$), and half the requests for A ($0.5 \cdot 3$), leading to a cache hit ratio of $4.5/7$, approximately 64% (an almost 50% improvement over the standard policy). This simple example motivates the significant potential of the approach, but also the need for a fundamental reconsideration of caching policies, even in relatively simple networking setups. Finally, while this simple example might tempt the reader to think that the new optimal policy is simply to (re-)rank contents based on *total* hit rate each can achieve (including SCHs), and then apply standard policies (e.g. picking the highest ranked ones), in fact we will show that the optimal policy is a hard combinatorial (cover) problem.

[1]A user that has already chosen a content might over-value her original choice and feel unhappy to swap it to an objectively equally interesting content. This effect is somewhat akin to the well-known *endowment effect* from Behavioral Economics [17].

C. Contributions

The main contributions of the paper are summarized as follows.

- *Soft Cache Hits (SCH) concept:* We introduce the novel concept of soft cache hits. To our best knowledge, this is the first time that this idea has been applied to edge caching for cellular networks (besides our own preliminary work [20]).
- *Soft Cache Hits (SCH) model:* We propose a generic model for mobile edge caching with soft cache hits that can capture a number of interesting content substitution scenarios (e.g. both Fig. 1(a) and Fig. 1(b)) and is versatile enough to apply on top of both non-cooperative (i.e. single cache) and cooperative caching frameworks [5], as well as various networking assumptions.
- *Analytical Investigation:* We prove that the problem of optimal edge caching with SCH is NP-hard even when considering a single cache only. This is in stark contrast to the standard case without SCH. We then prove that, despite the increased complexity, the generic problem of femto-caching with SCH still exhibits properties that can be taken advantage of to derive efficient approximation algorithms with provable performance.
- *Trace-based Validation:* We corroborate our SCH proposal and analytical findings through an extended evaluation on 5 real datasets containing information about related content, demonstrating that promising additional caching gains could be achieved in practice.

As a final remark, it is important to stress that *we do not propose to modify the recommendation systems themselves* (unlike [18], [19], for example). Instead, *our focus is on the caching policy side*, using the output of a state-of-art recommendation system for the respective content type as input to our problem (this will be further clarified in Section II). Of course, a content provider with a recommendation system can benefit from our approach to optimize its caching policies, or even modify its recommendations to incorporate soft cache hits. For example, upon peak hours, a content provider can carefully "steer" recommendations to optimize the network performance and user experience (e.g., from lower latency). Moreover, jointly optimizing both the caching and the recommendation sides of the problem could offer additional benefits. We defer this to future work. Overall, we believe that such a convergence between recommendation and caching systems is quite timely, given that dividing lines between Mobile Network Operators (MNO) and content providers are becoming more blurry, due to architectural developments like Multi-access Edge Computing (MEC) [14] and RAN Sharing [21].

In the following section we introduce the problem setup and our soft cache hits model corresponding to the example application of Fig. 1(a). In Section III we formulate and analyze the problem of edge caching with SCH for a single cache, and propose efficient caching algorithms. Then, in Section IV, we generalize the problem, analysis, and algorithms to the femto-caching case. In Section V we extend our model to capture scenarios as in the example application of Fig. 1(b), and show that our analytic findings are applicable to these scenarios as well. The performance evaluation is presented in Section VI. Finally, we discuss related work and future research directions in Section VII, and conclude our paper in Section VIII.

II. Problem Setup

A. Network and Caching Model

Network Model: Our network consists of a set of users $\mathcal{N}$ ($|\mathcal{N}| = N$) and a set of SCs (or, *helpers*) $\mathcal{M}$ ($|\mathcal{M}| = M$). Users are mobile and the SCs with which they associate might change over time. Since the caching decisions are taken in advance (e.g., the night before, as in [5], [6], or once per few hours or several minutes), it is hard to know the exact SC(s) each user will be associated at the time she requests a content. To capture user mobility, we propose a more generic model than the fixed bipartite graph of [5]:

$$q_{ij} \doteq Prob\{\text{user i in range of SC j}\},$$

or, equivalently, q_{ij} is the percentage of time a user i spends in the coverage of SC j. Hence, deterministic q_{ij} ($\in \{0,1\}$) captures the static setup of [5], while uniform q_{ij} ($q_{ij} = q, \forall i, j$) represents the other extreme (no advance knowledge).

Content Model: We assume each user requests a content from a catalogue $\mathcal{K}$ with $|\mathcal{K}| = K$ contents. A user $i \in \mathcal{N}$ requests content $k \in \mathcal{K}$ with probability p_k^i.[2] We will initially assume that all contents have the same size, and relax the assumption later.

Cache Model (Baseline): We assume that each SC/helper is equipped with storage capacity of C contents (all our proofs hold also for different cache sizes). We use the integer variable $x_{kj} \in \{0,1\}$ to denote if content k is stored in SC j. In the traditional caching model (baseline model), if a user i requests a content k which is stored in some nearby SC, then the content can be accessed directly from the local cache and a *cache hit* occurs. This type of access is considered "cheap", while a *cache miss* leads to an "expensive" access (e.g., over the SC backhaul and core network).

For ease of reference, the notation is summarized in Table I.

TABLE I: Important Notation

$\mathcal{N}$	set of users ($|\mathcal{N}| = N$)
$\mathcal{M}$	set of SCs / helpers ($|\mathcal{M}| = M$)
C	storage capacity of a SC
q_{ij}	probability user i in range of SC j
$\mathcal{K}$	set of contents ($|\mathcal{K}| = K$)
p_k^i	probability user i to request content k
x_{kj}	k is stored in SC j ($x_{kj} = 1$) or not ($x_{kj} = 0$)
u_{kn}^i	utility of content n for a user i requesting content k
$F_{kn}(x)$	distribution of utilities u_{kn}^i, $F_{kn}(x) = P\{u_{kn}^i \leq x\}$
u_{kn}	avg. utility for content pair $\{k, n\}$ (over all users)
s_k	size of content k

B. Soft Cache Hits

Up to this point the above model describes a baseline setup similar to the popular femto-caching framework [5]. The main departure in this paper is the following.

Related Content Recommendation: When a user consumes a content (or initially requests a content) that is not found in the local cache, we assume that an app proxy (e.g. YouTube, Netflix, Spotify), collocated or near this cache, looks at the list of contents its recommendation system deems related to the currently consumed content (or the initial request), checks which of them are available in the local cache, and recommends them to the user (see the example in Fig. 1(a)). If a user selects to consume next (or instead of the initial) one of them, a *(soft) cache hit* occurs, otherwise there is a cache miss and the network must fetch and deliver the original content.

Below, we first propose a soft cache hit model that captures the scenario of Fig. 1(a). We will use this model throughout Sections III and IV, to develop most of our theory. However, in Section V, we will modify our model to also analyze the scenario of Fig. 1(b), which we will refer to as *Related Content Delivery*.

Definition 1. *A user i that requests a content k that is not available, accepts a recommended content n with probability u_{kn}^i, where $0 \leq u_{kn}^i \leq 1$, and $u_{kk}^i = 1, \forall i, k$.*

These utilities/probabilities (in the remainder we use these terms interchangeably) define a content relation matrix $\mathbf{U^i} = \{u_{kn}^i\}$ for each user. They could be estimated from past statistics and/or user profiles, and are closely related to the output of the recommender for that user and that content app. For example, if a collaborative filtering algorithm suggested that the cosine distance [22] between files k and n for user i is 0.5, we could set $u_{kn}^i = 0.5$[3].

In some cases, the system might have a coarser view of these utilities (e.g., item-item recommendation [23]). We develop our theory and results for the most generic case of Definition 1,

[2]This generalizes the standard femto-caching model [5] which assumes same popularity per user. We can easily derive such a popularity p_k from p_k^i.

[3]Going from a content relation value to a value for the willingness of a user to accept that related content arguably entails some degree of subjectivity, given that this also depends on the amount and type of incentives offered to the user. Nevertheless, it is clear that whatever the actual value of u_{kn}^i, it will be some function of and positively correlated to underlying content relevance, which can be readily available from the respective recommendation system.

but we occasionally refer to the following two subcases, which might appear in practice:

Sub-case 1: The system does not know the exact utility u_{kn}^i for each node i, but only how they are distributed among all nodes, i.e., the distributions $F_{kn}(x) \equiv P\{u_{kn}^i \leq x\}$.

Sub-case 2: The system knows only the *average utility* u_{kn} per content pair $\{k, n\}$.

III. Single Cache with Soft Cache Hits

In order to better understand the impact of the related content matrices $\mathbf{U^i}$ on caching performance, we first consider a scenario where a user i is served by a single small cell, i.e., each user is associated to exactly one SC, but we might still not know in advance which. Such a scenario is in fact relevant in today's networks, where the cellular network first chooses a single SC to associate a user to (e.g., based on signal strength), and then the user makes its request [24]. In that case, we can optimize each cache independently. We can also drop the second index for both the storage variables x_{kj} and connectivity variables q_{ij}, to simplify notation.

In the remainder, we select the cache hit ratio (CHR) as the basic performance metric, similarly to the majority of the related work. However, the analysis for CHR maximization can be generalized to utility maximization [25], where "utility" can be the content access cost or delay, energy consumption, etc.

A. Soft Cache Hit Ratio

A request (from a user to a SC/helper) for a content $k \in \mathcal{K}$ would result in a (standard) cache hit only if the SC/helper stores the content k in its cache, i.e., if $x_k = 1$. Hence, the (baseline) *cache hit ratio* for this request is simply

$$CHR(k) = x_k$$

If we further allow for soft cache hits, the user might be also satisfied by receiving a different content $n \in \mathcal{K}$. The probability of this event is, by Definition 1, equal to u_{kn}^i. The following Lemma derives the total cache hit ratio in that case.

Lemma 1 (Soft Cache Hit Ratio (SCHR))**.** *Let $SCHR$ denote the expected cache hit ratio for a single cache (including regular and soft cache hits), among all users. Then,*

$$SCHR = \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1 - \prod_{n=1}^{K}\left(1 - u_{kn}^i \cdot x_n\right)\right). \quad (1)$$

Proof. The probability of satisfying a request for content k by user i with related content n is $P\{n|k,i\} = u_{kn}^i \cdot x_n$, since u_{kn}^i gives the probability of acceptance (by definition), and x_n denotes if content n is stored in the cache (if the content is not stored, then $P\{n|k,i\} = 0$). Hence, it follows easily that the probability of a *cache miss*, when content k is requested by user i, is given by[4] $\prod_{n=1}^{K}(1 - u_{kn}^i \cdot x_n)$. The complementary probability, defined as the *soft cache hit ratio* (SCHR), is then

$$SCHR(i, k, \mathbf{U}) = 1 - \prod_{n=1}^{K}(1 - u_{kn}^i \cdot x_n). \quad (2)$$

Summing up over all users that might be associated with that BS (with probability q_i) and all contents that might be requested (p_k^i) gives us Eq.(1). □

[4]To simplify our analysis, throughout our proofs we will assume that the user is informed about *all cached contents n with non-zero relevance u_{kn}^i to the original content k*. In practice, only a limited number of them would be recommended (e.g. the N most related among the cached ones, as in [19])). Our analysis also holds for this case, with limited modifications.

Lemma 1 can be easily modified for the sub-cases 1 and 2 of Definition 1 presented in Section II-B. We state the needed changes in Corollary 1.

Corollary 1. *Lemma 1 holds for the the sub-cases 1 and 2 of Definition 1, by substituting in the expression of Eq.* (1) *the term u_{kn}^i with*

$$u_{kn}^i \rightarrow E[u_{kn}^i] \equiv \int (1 - F_{kn}(x))\, dx \qquad \textit{(for sub-case 1)} \quad (3)$$

$$u_{kn}^i \rightarrow u_{kn} \qquad \textit{(for sub-case 2)} \quad (4)$$

Proof. The proof is given in Appendix A. □

B. Optimal SCH for Equal Content Sizes

The (soft) cache hit ratio depends on the contents that are stored in a SC/helper. The network operator can choose the storage variables x_k to maximize SCHR by solving the following optimization problem.

Optimization Problem 1. *The optimal cache placement problem for a single cache with soft cache hits and content relations described by the matrix $\mathbf{U^i} = \{u_{kn}^i\}$, $\forall i \in \mathcal{N}$, is*

$$\underset{X=\{x_1,\dots,x_K\}}{\textit{maximize}} \quad f(X) = \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1 - \prod_{n=1}^{K}\left(1 - u_{kn}^i \cdot x_n\right)\right) \quad (5)$$

$$s.t. \quad \sum_{k=1}^{K} x_k \leq C. \quad (6)$$

In the following, we prove that the above optimization problem is NP-hard (Lemma 2), and study the properties of the objective function Eq. (5) (Lemma 3) that allow us to design an efficient approximate algorithm (Algorithm 1) with provable performance guarantees (Theorem 1).

Lemma 2. *The Optimization Problem 1 is NP-hard.*

Lemma 3. *The objective function of Eq.(5) is submodular and monotone (non-decreasing).*

The proofs for the previous two Lemmas can be found in Appendices B and C, respectively.

We propose Algorithm 1 as a greedy algorithm for Optimization Problem 1: to select the contents to be stored in the cache, we start from an empty cache (line 1), and start filling it (one by one) with the content that increases the most the value of the objective function (line 4), till the cache is full. The computation complexity of the algorithm is $O\left(C \cdot K\right)$, since the loop (lines 2-6) denotes C repetitions, and in each repetition the objective function is evaluated y times, where $K \geq y \geq K - C + 1$. An efficient implementation of the step in line 4 can be based on the method of *lazy evaluations of the objective function*; due to space limitations, we refer the interested reader to [26].

Algorithm 1 $\left(1-\frac{1}{e}\right)$-approximation Greedy Algorithm for Optimization Problem 1.
computation complexity: $O\left(C\cdot K\right)$

```
    Input: utility $\{u^i_{kn}\}$, content demand $\{p^i_k\}$, mobility $\{q_i\}$, $\forall k,n\in\mathcal{K},\ i\in\mathcal{N}$
 1: $S_0 \leftarrow \emptyset;\ t\leftarrow 0$
 2: while $t < C$ do
 3:     $t\leftarrow t+1$
 4:     $n\leftarrow \underset{\ell\in K\setminus S_{t-1}}{argmax} f(S_{t-1}\cup\{\ell\})$
 5:     $S_t\leftarrow S_{t-1}\cup\{n\}$,
 6: end while
 7: $S^*\leftarrow S_t$
 8: return $S^*$
```

The following theorem gives the performance bound for Algorithm 1.

Theorem 1. *Let OPT be the optimal solution of the Optimization Problem 1, and S^* the output of Algorithm 1. Then, it holds that*

$$f(S^*)\geq\left(1-\frac{1}{e}\right)\cdot OPT \tag{7}$$

Proof. Lemma 3 shows that the Optimization Problem 1 belongs to the generic category of maximization of submodular and monotone functions (Eq. (5)) with a cardinality constraint (Eq. (6)). For such problems, it is known that the greedy algorithm achieves (in the worst case) a $\left(1-\frac{1}{e}\right)$-approximation solution [27], [26]. □

While the above is a strict worst case bound, it is known that greedy algorithms perform quite close to the optimal in most scenarios. In Sec. VI we show that this simple greedy algorithm can already provide interesting performance gains.

C. Optimal SCH for Different Content Sizes

Till now we have assumed that all contents have equal size. In practice, each content has a different size s_k and the capacity C of each cache must be expressed in Bytes. Additionally, if a user requests a video of duration X and she should be recommended an alternative one of similar duration Y (note that similar duration does not always mean similar size). While the latter could still be taken care of by the recommendation system (our study of a real dataset in Sec. VI suggests that contents of different sizes might still be tagged as related), we need to revisit the optimal allocation problem: the capacity constraint of Eq.(6) is no longer valid, and Algorithm 1 can perform arbitrarily bad [26].

Optimization Problem 2. *The optimal cache placement problem for a single cache with soft cache hits and variable content sizes, and content relations described by the matrix $\mathbf{U^i}=\{u^i_{kn}\}$, $\forall i\in\mathcal{N}$, is*

$$\underset{X=\{x_1,\ldots,x_K\}}{maximize}\quad f(X)=\sum_{i=1}^{N}\sum_{k=1}^{K}p^i_k\cdot q_i\cdot\left(1-\prod_{j=1}^{K}\left(1-u^i_{kn}\cdot x_n\right)\right) \tag{8}$$

$$s.t.\quad \sum_{k=1}^{K}s_k x_k\leq C. \tag{9}$$

Remark: Note that the objective is still in terms of cache hit ratio, and does not depend on content size. This could be relevant, e.g., when the operator is doing edge caching to reduce *access latency* to contents (latency is becoming a core requirement in 5G).

Algorithm 2 $\frac{1}{2}\cdot\left(1-\frac{1}{e}\right)$-approximation Algorithm for Optimization Problem 2.
computation complexity: $O\left(K^2\right)$

```
    Input: utility $\{u^i_{kn}\}$, content demand $\{p^i_k\}$, content size $\{s_k\}$, mobility $\{q_i\}$, $\forall k,n\in\mathcal{K},\ i\in\mathcal{N}$
 1: $S^{(1)}\leftarrow$ModifiedGreedy($\emptyset$,[$s_1$, $s_2$,...,$s_k$])
 2: $S^{(2)}\leftarrow$ModifiedGreedy($\emptyset$,[1, 1,...,1])
 3: if $f(S^{(1)})>f(S^{(2)})$ then
 4:     $S^*\leftarrow S^{(1)}$
 5: else
 6:     $S^*\leftarrow S^{(2)}$
 7: end if
 8: return $S^*$

 9: function ModifiedGreedy($S_0$,[$w_1$, $w_2$,...,$w_k$])
10:     $\mathcal{K}^{(1)}\leftarrow\mathcal{K};\ c\leftarrow 0;\ t\leftarrow 0$
11:     while $\mathcal{K}^{(1)}\neq\emptyset$ do
12:         $t\leftarrow t+1$
13:         $n\leftarrow \underset{\ell\in K\setminus S_{t-1}}{argmax}\frac{f(S_{t-1}\cup\{\ell\})}{w_\ell}$
14:         if $c+w_n\leq C$ then
15:             $S_t\leftarrow S_{t-1}\cup\{n\}$
16:             $c\leftarrow c+w_n$
17:         else
18:             $S_t\leftarrow S_{t-1}$
19:         end if
20:         $\mathcal{K}^{(1)}\leftarrow\mathcal{K}^{(1)}\setminus\{n\}$
21:     end while
22:     return $\leftarrow S_t$
23: end function
```

The problem is a set cover problem variant with a knapsack type constraint. We propose the approximation Algorithm 2 for this problem, which is a "fast greedy" algorithm (based on a modified version of the greedy Algorithm 1) and has complexity $O\left(K^2\right)$.

Theorem 2.
(1) The Optimization Problem 2 is NP-hard.
(2) Let OPT be the optimal solution of the Optimization Problem 2, and S^* the output of Algorithm 2. Then, it holds that

$$f(S^*)\geq\frac{1}{2}\left(1-\frac{1}{e}\right)\cdot OPT \tag{10}$$

Proof. A sketch of the proof can be found in Appendix D. □

In fact, a polynomial algorithm with better performance $\left(1-\frac{1}{e}\right)$-approximation could be described, based on [28]. However, the improved performance guarantees come with a significant increase in the required computations, $O\left(K^5\right)$, which might not be feasible in a practical scenario when the catalog size K is large. We therefore just state its existence, and do not consider the algorithm further in this paper (the algorithm can be found in [29]).

IV. Femtocaching with Related Content Recommendation

Building on the results and analytical methodology of the previous section for the optimization of a single cache with soft cache hits, we now extend our setup to consider the complete problem with cache overlaps (referred to as "femtocaching" [5]). Note, however, that we *do* consider user mobility, through variables q_{ij}, unlike previous works in this framework that often assume static users. Due to space limitations, we focus on the case of fixed content sizes.

In this scenario, a user $i \in \mathcal{N}$ might be covered by more than one SCs/helpers $j \in \mathcal{M}$, i.e. $\sum_j q_{ij} \geq 1, \forall i$. A user is satisfied, if she receives the requested content k or *any* other related content (that she will accept), from *any* of the SCs/helpers within range. Hence, similarly to Eq. (2), the total cache hit ratio SCHR (that includes regular and soft cache hits) is written as

$$SCHR(i,k,\mathbf{U}) = 1 - \prod_{j=1}^{M}\prod_{n=1}^{K}\left(1 - u_{kn}^{i} \cdot x_{nj} \cdot q_{ij}\right) \quad (11)$$

since for a cache hit a user i needs to be in the range of a SC j (term q_{ij}) that stores the content n (term x_{nj}), and accept the recommended content (term u_{kn}^{i}).

Considering (i) the request probabilities p_k^i, (ii) every user in the system, and (iii) the capacity constraint, gives us the following optimization problem.

Optimization Problem 3. *The optimal cache placement problem for the femtocaching scenario with soft cache hits and content relations described by* $\mathbf{U^i} = \{u_{kn}^i\}$, $\forall i \in \mathcal{N}$, *is*

$$\underset{X=\{x_{11},\dots,x_{KM}\}}{\text{maximize}} \; f(X) =$$

$$= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot \left(1 - \prod_{j=1}^{M}\prod_{n=1}^{K}\left(1 - u_{kn}^{i} \cdot x_{nj} \cdot q_{ij}\right)\right), \quad (12)$$

$$s.t. \quad \sum_{k=1}^{K} x_{kj} \leq C, \quad \forall j \in \mathcal{M}. \quad (13)$$

The following lemma states the complexity of the above optimization problem, as well as its characteristics that allow us to design an efficient approximation algorithm.

Lemma 4.
(1) The Optimization Problem 3 is NP-hard,
(2) with submodular and monotone (non-decreasing) objective function (Eq. (12)*) and a matroid constraint (Eq.* (13)*).*

Proof. We prove Lemma 4 by extending the basic ideas of the single-cache case, and following a similar methodology as in the proofs of Lemmas 2 and 3; the detailed proof is given in [29]. □

Lemma 4 states that the Optimization Problem 3 is a maximization problem with a submodular function and a matroid constraint. For this type of problems, a greedy algorithm can guarantee an $\frac{1}{2}$-approximation of the optimal solution [26]. The greedy algorithm is similar to Algorithm 1 and is of computational complexity $O\left(K^2M^2\right)$; i.e., instead of considering only contents in the allocation, now tuples {content, helper} need to be greedily allocated until the caches of helpers are full (for a detailed pseudocode of the algorithm, we refer the reader to [29]).

Theorem 3. *Let OPT be the optimal solution of the Optimization Problem 3, and S^* the output of the greedy algorithm. Then, it holds that*

$$f(S^*) \geq \frac{1}{2} \cdot OPT \quad (14)$$

Submodular optimization problems have received considerable attention recently, and a number of sophisticated approximation algorithms have been considered (see, e.g., [26] for a survey). For example, a better $\left(1 - \frac{1}{e}\right)$-approximation (with increased computation complexity though) can be found following the "multilinear extension" approach [30], based on a continuous relaxation and pipage rounding. A similar approach has also been followed in the original femtocaching paper [5]. Other methods also exist that can give an $\left(1 - \frac{1}{e}\right)$-approximation [31]. Nevertheless, minimizing algorithmic complexity or optimal approximation algorithms are beyond the scope of this paper. Our goal instead is to derive fast and efficient algorithms (like greedy) that can handle the large content catalogues and content related graphs $\mathbf{U}$, and compare the performance improvement offered by soft cache hits. The worst-case performance guarantees offered by these algorithms are added value.

V. Femtocaching with Related Content Delivery

We have so far considered a system corresponding to the example of Fig. 1(a), where a cache-aware system *recommends* alternative contents to users (in case of a cache miss), but users might not accept them. In this section, we consider a system closer to our second example of Fig. 1(b), where the system *delivers* some related content that is locally available *instead of the original content*, in case of a cache miss. While a more extreme scenario, we believe this might still have application in a number of scenarios, as explained in Section I (e.g., for low rate plan users under congestion, or in limited access scenarios [32], [33]).

In the following, we model the related content delivery system, formulate the respective optimization problem, and show that it has the same properties with the problems in the previous sections, which means that our results and algorithms apply to this context as well. We present only the more generic femto-cache case of Sec. IV; the analysis and results for the single cache cases of Sec. III follow similarly.

Since now original requests might not be served, the (soft) cache hit ratio metric does not describe sufficiently the performance of this system. To this end, we modify the definition of content utility:

Definition 2. *When a user i requests a content k that is not locally available and the content provider delivers an alternative content n then the user satisfaction is given by the utility u_{kn}^i. $u_{kn}^i \in \mathbb{R}$ is a real number, and does not denote a probability of acceptance, but rather the happiness of user i when she receives n instead of k. Furthermore $u_{kk}^i = U_{max}, \forall i$.*

Note: we stress that the utilities u_{kn}^i in Definition 2 do not represent the probability a user i to accept a content n (as in Definition 1), but the satisfaction of user i given that she accepted content n. User satisfaction can be estimated by past statistics, or user feedback, e.g., by asking user to rate the received alternative content.

Let us denote as $G_i(t) \subseteq \mathcal{M}$ the set of SCs with which the user i is associated at time t. Given Definition 2, when a user i requests at time t a content k that is not locally available, we assume a system (as in Fig. 1(b)) that delivers to the user the cached content with the *highest* utility[5], i.e., the content n where

$$n \equiv \arg\max_{\ell \in \mathcal{K}, j \in G_i(t)} \left\{u^i_{k\ell} \cdot x_{\ell j}\right\} \tag{15}$$

Hence, the satisfaction of a user i upon a request for content k is

$$\max_{n \in \mathcal{K}, j \in G_i(t)} \left\{u^i_{kn} \cdot x_{nj}\right\} \tag{16}$$

Using the above expression and proceeding similarly to Section IV, we formulate the optimization problem that the network needs to solve to optimize the total user satisfaction (among all users and all content requests), which we call *soft cache hit user satisfaction* (SCH-US).

Optimization Problem 4 (SCH-US)**.** *The optimal cache placement problem for the femtocaching scenario with related content delivery and content relations described by the matrix* $\mathbf{U} = \{u^i_{kn}\}$ *is*

$$\underset{X=\{x_1,\dots,x_K\}}{\textit{maximize}} \quad f(X) =$$
$$= \sum_{i=1}^{N} \sum_{k=1}^{K} p^i_k \cdot E_{G_i}\left[\max_{n \in \mathcal{K}, j \in \mathcal{M}} \left(u^i_{kn} \cdot x_{nj} \cdot Q_{ij}\right)\right], \tag{17}$$
$$\textit{s.t.} \quad \sum_{k=1}^{K} x_{kj} \le C, \quad \forall j \in \mathcal{M}. \tag{18}$$

where $Q_{ij} = \begin{cases} 1 & \text{, if } j \in G_i \\ 0 & \text{, otherwise} \end{cases}$ *, and the expectation* $E_{G_i}[\cdot]$ *is taken over the probabilities* $P\{G_i\} = \prod_{j \in G_i} q_{ij} \cdot \prod_{j \notin G_i} (1 - q_{ij})$*.*

For the sub-cases 1 and 2 of Definition 1 presented in Sec. II-B, the following corollary holds.

Corollary 2. *The expression of Eq.* (17) *needs to be modified as*

$$E_{G_i}\left[\max_{n \in \mathcal{K}, j \in \mathcal{M}} \left(u^i_{kn} \cdot x_{nj} \cdot Q_{ij}\right)\right] \rightarrow E_{G_i}\left[\max_{n \in S} \left(u^i_{kn}\right)\right] =$$
$$= E_{G_i}\left[\int \left(1 - \prod_{n \in S} F_{kn}(x)\right) dx\right] \tag{19}$$
$$u^i_{kn} \rightarrow u_{kn} \tag{20}$$

where $S = \{\ell : \ell \in \mathcal{K}, m \in \mathcal{M}, x_{\ell m} \cdot Q_{im} = 1\}$*, for the sub-cases 1 and 2 of Definition 1, respectively.*

Proof. Due to space limitation the proof is given in [29]. □

We now prove the following Lemma, which shows that Theorem 3 applies also to the Optimization Problem 4, and thus it can be efficiently solved by the same greedy algorithm (where the objective function of Eq. (17) is now used).

Lemma 5.
(1) The Optimization Problem 4 is NP-hard,
(2) with submodular and monotone objective function (Eq. (17)*).*

[5]Equivalently, the system can recommend all the stored contents to the user and then allow the user to select the content that satisfies her more.

Proof. The proof is given in Appendix E. □

VI. Performance Evaluation

In this section, we investigate the gains of employing soft cache hits and the performance of the proposed algorithms. We first analyze 5 real datasets collected from different content-centric applications/sources, such as YouTube and Amazon-TV, as well as other types of contents (e.g. Android applications) or data sources (like MovieLens) (Sec. VI-A). We have also tested our schemes with some data related to personalized radio (lastFM) with similar conclusions. The datasets contain information about content relations, based on which we build the utility matrices $\mathbf{U}$. We use these realistic utility matrices $\mathbf{U}$ in our simulations to study the performance of caching with or without soft cache hits. In Sec. VI-B we describe the simulation setup, and present and discuss the results in Sec. VI-C.

A. Datasets of Content Relations

YouTube dataset. We consider a dataset of YouTube videos from [34][6]. The dataset contains several information about the videos, such as their popularity, size, and a list of related videos (as recommended by YouTube). We build the utility matrix $\mathbf{U} = \{u_{nk}\}$, where $u_{nk} = 1$ if video n is in the list of related videos of k (or vice-versa), and otherwise $u_{nk} = 0$.

Amazon datasets. We also analyze 3 datasets of product reviews from Amazon [35] for Android applications (*Amazon-App*), Movies and TV (*Amazon-TV*), and Videogames (*Amazon-VG*). The datasets include for each item a list of contents that are "also bought". [7] We consider for each dataset 10000 of its items, and build a utility matrix $\mathbf{U} = \{u_{nk}\}$, where $u_{nk} = 1$ if item n is also bought with item k (or vice-versa), and otherwise $u_{nk} = 0$.

MovieLens dataset. We finally consider a movies-rating dataset from the MovieLens website [36], containing 69162 ratings (from 0.5 stars to 5) of 671 users for 9066 movies. As these datasets contain only raw user ratings and not movie relations per se, to obtain content relation matrix U, in this case, we do an intermediate step and apply a standard concept from collaborative filtering [22]. Specifically, we calculate the similarity of each pair of contents based on their common ratings as their cosine-distance metric:

$$sim(n,k) = \frac{\sum_{i=1}^{\#users} r_i(n) \cdot r_i(k)}{\sqrt{\sum_{i=1}^{\#users} r_i^2(n)} \cdot \sqrt{\sum_{i=1}^{\#users} r_i^2(k)}}$$

where we normalized the ratings r_i, by subtracting from each rating the average rating of that item, so that we obtain similarity values $\in [-1, 1]$. Due to the sparsity of the dataset (few common ratings), we also apply an item-to-item collaborative filtering (using 10 similar items) in order to predict the missing user ratings per item, and thus the missing similarity values. We build the utility matrix $\mathbf{U} = \{u_{nk}\}$ with $u_{nk} = \max\{0, sim(n,k)\}$, i.e, $u_{nk} \in [0,1]$. Finally,

[6]Data from 27-07-2008, and depth of search up to 3; see details in [34]

[7]Our main motivation to use the game and app datasets was to also validate the robustness of our approach to other types of data. Soft cache hits though might be more relevant for free apps or games, rather than paid products.

TABLE II: Information contained in datasets.

	content popularity	content size	content relations u_{kn} $\in \{0,1\}$	$\in [0,1]$
Amazon-*	×	×	✓	×
MovieLens	✓	×	×	✓
YouTube	✓	✓	✓	×

TABLE III: Dataset analysis.

	#contents	content relations $E[R]\left(\frac{std[R]}{E[R]}\right)$	popularity $E[p]\left(\frac{std[p]}{E[p]}\right)$
Amazon-App	8229	16.0 (2.2)	-
Amazon-TV	2789	7.8 (1.0)	-
Amazon-VG	5614	22.0 (1.1)	-
MovieLens	4622	125.8 (0.5)	15 (1.6)
YouTube	2098	5.3 (0.7)	500 (3.1)

we assign to each item a popularity value equal to the number of ratings for this item.

For ease of reference, Table II presents the information contained in each dataset.

Due to the sparsity of the YouTube dataset, we only consider contents belonging to the largest connected component (defining as adjacencies, the positive entries of the utility matrix). For consistency, we consider only the contents in the largest connected component for the other datasets as well. Moreover, since the Amazon and YouTube datasets do not contain per-user information, and the per-user data in the MovieLens dataset is sparse, we consider the sub-case-2 of Definition 1, i.e., $u_{kn}^i = u_{kn}$ for all users i.

The number of remaining contents for each dataset are given in Table III. We also calculate for each content the number of its related contents $R_n = \sum_k u_{nk}$ (or the sum of its utilities for the MovieLens dataset where $u_{kn} \in [0,1]$), and present the corresponding statistics in Table III along with the statistics for the content popularity.

B. Simulation Setup

Cellular network. We consider an area of 1 km^2 that contains M SCs. SCs are randomly placed in the area (following a Poisson point process), which is a common assumption in related work [5], [37]. An SC can serve a request from a user, when the user is inside its communication range, which we set to 200 meters. We also consider N mobile users.

We select as default parameters: $N = 50$ users, and $M = 20$ SCs with caching capacity $C = 5$ (contents). This creates a relatively dense network, where a random user is connected to 3 SCs *on average*.

Content demand. We consider a scenario of content demand for each dataset of Sec. VI-A, with the corresponding set of contents, content popularities and relations (utility matrix). For datasets without information on content popularity (see Table II), we generate a random sample of popularity values drawn from a Zipf distribution in $[1, 400]$[8] with exponent $\alpha = 2$. For each scenario we generate a set of 20 000 requests according to the content popularity, over which we average our results. When soft cache hits are allowed, we assume the *related content recommendation* model of Definition 1 (see also Fig. 1(a)).

[8]The max value is selected equal to the max number of requests per user, i.e., $\frac{\#requests}{\#contents}$.

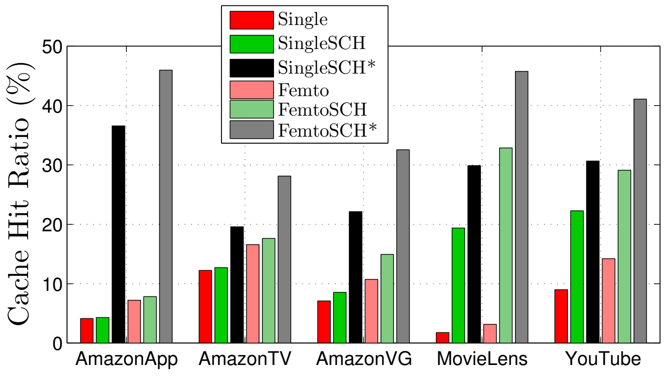


Fig. 2: Cache hit ratio for all datasets and caching schemes, for the default scenario.

Unless otherwise stated, the simulations use the default parameters summarized in Table IV.

Caching schemes / algorithms. We consider and compare the following schemes for single-SC (*single*) and multi-SC (*femto*) to user association.

- *Single*: A single cache accessible per user (e.g., the closest one). Only normal cache hits allowed, and the most popular contents are stored in each cache, which is the optimal policy in this simple setup. It will serve as the baseline for single cache scenarios.
- *SingleSCH*: Here soft cache hits are allowed. However, the caching policy is still based on popularity as before (i.e., is not explicitly optimized to exploit SCHs)
- *SingleSCH**: This is our proposed policy. Here soft cache hits are allowed, *and* the caching policy is optimized to fully exploit this (according to Algorithm 1 or Algorithm 2).
- *Femto*: Femto-caching without soft cache hits. This is the baseline scheme for this class of scenarios, where the proposed algorithm from [5] is applied.
- *FemtoSCH*: Femto-caching based content placement (same as in *Femto)*, but allowing soft cache hits on user requests (a posteriori).
- *FemtoSCH**: Our proposed policy. Femto-caching is explicitly optimized for soft cache hits, according the greedy algorithm (Sec. IV).

C. Results

1. Overall performance

We simulate scenarios for all datasets / utility matrices with the default parameters (Table IV), both under single and multi user-SC association. Fig. 2 shows the achieved cache hit ratio CHR (or soft cache hit ratio, SCHR) under the baseline caching (*Single/SingleSCH/Femto/FemtoSCH*) and the SCHR under a content placement using our algorithms (*SingleSCH*/FemtoSCH**).

Key Message: *Allowing soft cache hits can lead to a dramatic increase in the cache hit ratio.*

Comparing the cache hit ratio (CHR) under the popularity-based caching (*Single/Femto* - red/pink bars) and the schemes we propose (*SingleSCH*/FemtoSCH** - black/grey bars), shows that allowing soft cache hits brings a significant increase in the CHR for all datasets. The relative gain ranges from 60%

TABLE IV: Parameters used in simulations: default scenario.

Parameter	Value	Parameter	Value
Area	1 x 1 km	Cache size, C	5
nb. of SCs, M	20	nb. of users N	50
SC comm. range	200 m	Zipf distr.	$\in [1, 400]$, $\alpha = 2$

in the *Amazon-TV* case, up to around 780% in the *Amazon-App* case, for the *Single* scenarios; the relative gains in the *Femto* scenarios are similarly impressing (from 70% up to 530%, respectively). These initial results indicate that soft cache hits can be a promising solution for future mobile networks, by increasing the virtual capacity of mobile edge caching.

Key Message: ***While gains can sometimes already be achieved just by allowing soft cache hits, to fully take advantage of soft cache hits, the caching policy should be redesigned to explicitly take these into account (through the utility matrix).***

Fig. 2 demonstrates that gains could already be achieved by simply introducing soft cache hits on top of existing (state-of-the-art) caching policy (*SingleSCH/FemtoSCH* - dark/light green bars), but these are scenario-dependent. For example, in the *Amazon* scenarios the increase in CHR by allowing soft cache hits is marginal (red vs. green bars), while in the *YouTube* scenario it is $1.5\times$ higher. In contrast, explicitly designing the caching policy to exploit soft cache hits allows for important gains in all scenarios (black/grey bars). Specifically, in the *Amazon* scenarios the performance gains are almost entirely due to the caching algorithm (just allowing soft cache hits, does not improve performance), while in the *YouTube* scenario our utility-aware algorithms outperform by around 40% popularity-based caching. These results show clearly that existing caching policies are not capable to exploit the full potential of soft cache hits.

2. *Impact of network parameters*

We proceed to study the effect of network parameters, on the performance of soft cache hits schemes. We consider the *YouTube* dataset, for which the soft cache hits schemes (*SingleSCH*/FemtoSCH**) have a moderate gain (around $1.5-3\times$) over the baseline schemes. We simulate scenarios where we vary the cache size C and the number of SCs M; the remaining parameters are set as in the default scenario (Table IV).

Key Message: ***(a) Soft cache hits improve performance irrespectively of the underlying network parameters (even for a few SCs with small capacity); (b) Combining femto-caching and soft cache hits achieves significantly higher CHR that today's solutions.***

Cache size impact: We first investigate the impact of cache size, assuming fixed content sizes. Fig. 3 depicts the total cache hit ratio, for different cache sizes C: we consider a cache size per SC between 2 and 15 contents. The simulations suggest that the *SingleSCH*/FemtoSCH** scenarios consistently achieve more than $2.5\times$ (single) and $1.2\times$ (femto) higher CHR than *Single/Femto*. What is more, these gains are applicable to both single- and femto- caching. The two methods (femto-caching and soft cache hits) together offer a total of $3.3\times$ to $7\times$ improvement compared to the baseline scenario *Single*. Finally, even with a cache size per SC of about 0.1% of the

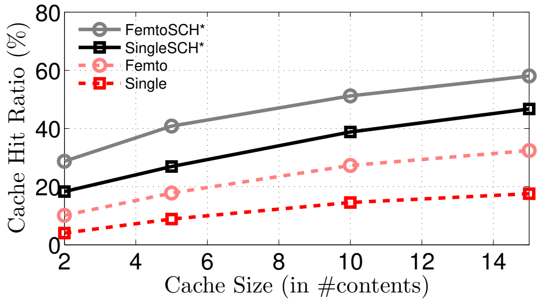


Fig. 3: Cache hit ratio vs. cache size C.

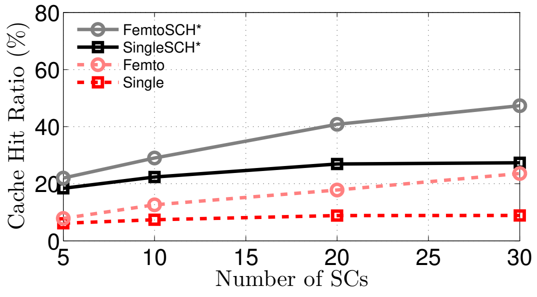


Fig. 4: Cache hit ratio vs. number of SCs M.

total catalog ($C = 2$), introducing soft cache hits offers 29% CHR (*SingleSCH**), whereas today's practices (popularity-based caching without SCH) would achieve only 4% CHR.

SC density impact: In Fig. 4 we consider the impact of SC density. In sparse scenarios (e.g., $M = 5$), a user usually is in the range of at most one SC. For this reason, Femto and Single perform similarly. As the SC density increases, the basic *Femto* is able to improve performance, as expected, by exploiting cache cooperation. However, every $2\times$ increase in density, which requires the operator doubling the infrastructure cost, offers roughly a relative improvement of $30-50\%$. Simply introducing soft cache hits instead, suffices to provide a $2\times$ improvement.

Key Message: ***The extra cost to incentivize soft cache hits might be quite smaller than the CAPEX/OPEX costs in infrastructure investment to achieve comparable performance gains.***

3. *Impact of utility matrix*

We further investigate the impact of the content relations as captured by the matrix $\mathbf{U}$ (and its structure). To quantify the content relations, we use as a metric the sum of the utilities per content $R_n = \sum_k u_{nk}$ (see also Sec. VI-A and Table III).

Key Message: ***The CHR increases with the density*** $(E[R])$ ***of the utility matrix. Even low utility values*** u_{kn} ***can significantly contribute towards a higher CHR.***

The first important parameter to consider is the average value of R_n, i.e., the density of the utility matrix $\mathbf{U}$. We consider the *MovieLens* dataset, where the utilities u_{kn} are real numbers in the range $[0, 1]$. To investigate the impact of the density of $\mathbf{U}$, we consider scenarios where we vary the matrix $\mathbf{U}$: for each scenario we set a threshold u_{min} and take into account only the relations between contents with utility

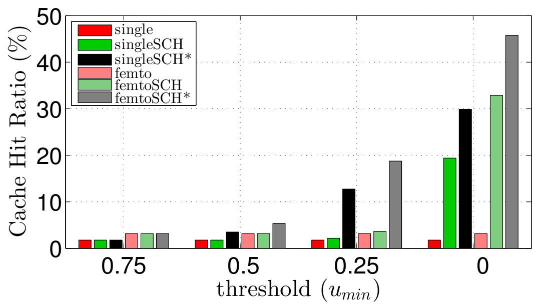


Fig. 5: Cache hit ratio for the MovieLens dataset for scenarios with different u_{min} thresholds; default scenario.

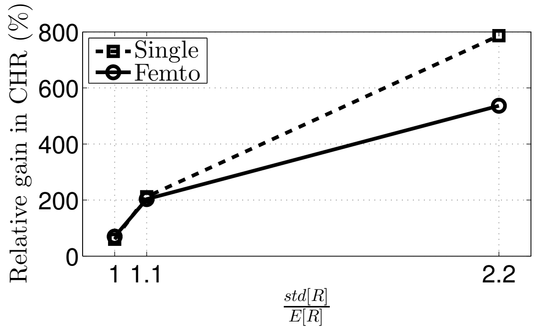


Fig. 6: Relative increase in the cache hit ratio due to soft cache hits (y-axis). *Amazon* scenarios with different variance of number of related contents (x-axis).

higher than u_{min}, i.e.,

$$\mathbf{U}' = \{u'_{kn}\} = \begin{cases} u_{kn} & \text{if } u_{kn} \geq u_{min} \\ 0 & \text{if } u_{kn} < u_{min} \end{cases}$$

Table V gives the density of the utility matrix for the different values of the threshold u_{min}, and Fig. 5 shows the cache hit ratio for these scenarios. When u_{min} is set to a large value (i.e., only few relations are taken into account and the matrix $\mathbf{U}$ is sparse), there is no ($u_{min} = 0.75$) or negligible ($u_{min} = 0.5$) improvement from soft cache hits. However, as the density of $\mathbf{U}$ increases (lower thresholds u_{min}), the gains in CHR significantly increase; note that these gains are from content relations with low utility values (i.e., less than 0.5 or 0.25 for the two rightmost scenarios, respectively). Moreover, it is interesting that in the scenario with $u_{min} = 0.25$, the gains are almost entirely due to the more efficient caching from our algorithms, i.e., popularity-based caching would not be efficient even if soft cache hits were allowed (green bars).

TABLE V: Utility matrix density for the MovieLens dataset for different u_{min} thresholds.

threshold u_{min}	0.75	0.5	0.25	0
$E[R]$	0.9	10.8	49.0	125.8

Key Message: ***Highly skewed distributions of #relations, R_n, can lead to more efficient caching (in analogy to heavy tailed popularity distributions).***

Our simulation study demonstrates the effect of the variance of the number of related contents, i.e., $\frac{std[R]}{E[R]}$. We consider the three *Amazon* scenarios of Fig. 2 that have the same content popularity distribution and similar $E[R]$ values (see Sec. VI-B). Fig. 6 shows the relative gain (of soft cache schemes over baseline schemes) in these scenarios where the distribution of R_n has different variance (see Table III). When the variance is very high (*Amazon-App*, $\frac{std[R]}{E[R]} = 2.2$) the CHR under soft cache hit schemes is almost an order of magnitude larger than the baseline scenarios. Large variance means that a few contents have very high R_n; thus storing these contents allows to serve requests for a large number of other (non-cached) contents as well. Finally, an interesting observation is that the variance of the distribution plays a more important role than the density of the utility matrix: although the utility matrix in the *Amazon-VG* scenario ($E[R] = 22$, $\frac{std[R]}{E[R]} = 1.1$) is denser than in the *Amazon-App* scenario ($E[R] = 16$, $\frac{std[R]}{E[R]} = 2.2$), the gain of the latter is higher due to the higher variance.

4. Performance of scheme extensions

Key Message: ***The gain of our algorithms is consistent for all the considered variations of soft cache hits scenarios.***

Finally, we evaluated scenarios with (a) contents of different size and (b) *related content delivery* model (Def. 2), and observed the following. In the former scenarios (from the *YouTube* dataset), the performance improves considerably for all cache size values (e.g., similarly to the equal content sizes case). In the latter scenarios (from the *MovieLens* dataset; similar to scenarios of Fig. 5), the *user satisfaction* significantly increases with related content delivery (i.e., soft cache hits), and denser matrices (i.e., higher willingness of users to accept related contents) lead to better performance.

VII. Related Work

Mobile Edge Caching. Densification of cellular networks, overlaying the standard macro-cell network with a large number of SCs (e.g., pico- or femto-cells), has been extensively studied and is considered a promising solution to cope with data demand [38], [39], [40]. As this densification puts a tremendous pressure on the backhaul network, researchers have suggested storing popular content at the "edge", e.g., at SCs [5], user devices [8], [9], [12], or vehicles [10], [11].

Our work is complementary to these approaches: it can utilize such mobile edge caching systems and further optimize the cache allocation when there is a cache-aware recommender systems in place. We have applied this approach in the context of mobile (ad-hoc) networks with delayed content delivery [41] as well, and applied it here for the first time in the context of femto-caching [5]. Additional research directions have also recently emerged, more closely considering the interplay between caching and the physical layer such as Coded Caching [7] and caching for coordinated (CoMP) transmission [42], [43]. We believe the idea of soft cache hits could be applied in these settings as well, and we plan to explore this as future work.

Caching and Recommendation Interplay. There exist some recent works that have jointly considered caching and recommendation for wireless systems [19], peer-to-peer networks [44], and CDNs [45], [18]. Specifically, [44] studies the interplay between a recommendation system and the performance of content distribution on a peer-to-peer network like

BitTorrent (e.g., recommending contents based on the number of "seeders") towards improving performance.

[45] shows that users tend to follow YouTube's suggestions, and despite the large catalog of YouTube, the top-10 recommendations are usually common for different users in the same geographical region. Hence, CDNs can use the knowledge from the recommendation system to improve their content delivery. Finally, [18], [19] propose approaches of recommended list reordering, which can achieve higher cache hit ratios (e.g., for YouTube servers/caches).

These works, except for [19] which is closer to our study, (i) focus on the recommendation side of the problem, ignoring or simplifying the optimal caching algorithm, and (ii) do not consider the wireless cooperative caching aspect of the problem. Nevertheless, the increasing dependence of user requests on the output of recommender systems clearly suggests that there is an opportunity to further improve the performance of (mobile) edge caching by jointly optimizing both, with minimum impact on user Quality of Experience.

A preliminary version of our work appears in [20]. However, there are a number of key additions in this work: (i) we consider in detail the single-cache scenario (Sec. III) and propose a more efficient algorithm (Alg. 1), as well as an algorithm for unequal content sizes (Alg. 2); (ii) we introduce the entirely new model of Sec. V and the corresponding analysis; (iii) we collected and analyzed 4 extra real datasets of content relations (*Amazon* and *MovieLens* datasets), in order to extend the evaluation and provide further insights on the gains of SCH (Sec. VI).

VIII. Conclusions

In this paper, we have proposed the idea of *soft cache hits*, where an alternative content can be recommended to a user, when the one she requested is not available in the local cache. While normal caching systems would declare a cache miss in that case, we argue that an appropriate recommended content, related to the original one can still satisfy the user with high enough probability. We then used this idea to design such a system around femto-caching, and demonstrated that considerable additional gains, *on top of those of femto-caching* can be achieved using realistic scenarios and data. We believe this concept of soft cache hits has wider applicability in various caching systems.

References

[1] F. Boccardi, R. Heath, A. Lozano, T. Marzetta, and P. Popovski, "Five Disruptive Technology Directions for 5G," *IEEE Comm. Mag. SI on 5G Prospects and Challenges*, 2014.

[2] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *Proc. IEEE INFOCOM*, pp. 1–9, 2010.

[3] M. Leconte, G. Paschos, L. Gkatzikis, M. Draief, S. Vassilaras, and S. Chouvardas, "Placing dynamic content in caches with small population," in *Proc. IEEE Infocom*, 2016.

[4] S. Elayoubi and J. Roberts, "Performance and cost effectiveness of caching in mobile access networks," in *Proceedings of International Conference on Information-Centric Networking, ICN '15, San Francisco, USA,*, 2015.

[5] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *Proc. IEEE INFOCOM*, 2012.

[6] 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, 2014.

[7] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. on Information Theory*, vol. 60, pp. 2856–2867, May 2014.

[8] P. Sermpezis and T. Spyropoulos, "Effects of content popularity on the performance of content-centric opportunistic networking: An analytical approach and applications," *IEEE/ACM Transactions on Networking*, vol. 24, no. 6, pp. 3354–3368, 2016.

[9] B. Han, P. Hui, V. S. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan, "Mobile data offloading through opportunistic communications and social participation," *IEEE Trans. on Mobile Computing*, 2012.

[10] J. Whitbeck, M. Amorim, Y. Lopez, J. Leguay, and V. Conan, "Relieving the wireless infrastructure: When opportunistic networks meet guaranteed delays," in *Proc. IEEE WoWMoM*, 2011.

[11] L. Vigneri, T. Spyropoulos, and C. Barakat, "Storage on Wheels: Offloading Popular Contents Through a Vehicular Cloud," in *Proc. IEEE WoWMoM*, 2016.

[12] P. Sermpezis and T. Spyropoulos, "Offloading on the edge: Performance and cost analysis of local data storage and offloading in HetNets," in *Proc. IEEE WONS*, pp. 49–56, 2017.

[13] R. Zhou, S. Khemmarat, and L. Gao, "The impact of youtube recommendation system on video views," in *In Proc. of ACM IMC 2010*.

[14] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing a key technology towards 5g," *ETSI White Paper No. 11*, 2016.

[15] "T-Mobile Music Freedom." https://www.t-mobile.com/offer/free-music-streaming.html, 2017.

[16] Y. Yiakoumis, S. Katti, and N. McKeown, "Neutral net neutrality," in *Proc. ACM SIGCOMM*, pp. 483–496, 2016.

[17] D. Kahneman. New York: Farrar, Straus and Giroux, 2011.

[18] D. K. Krishnappa, M. Zink, C. Griwodz, and P. Halvorsen, "Cache-centric video recommendation: an approach to improve the efficiency of youtube caches," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 11, no. 4, p. 48, 2015.

[19] L. E. Chatzieleftheriou, M. Karaliopoulos, and I. Koutsopoulos, "Caching-aware recommendations: Nudging user preferences towards better caching performance," in *Proc. IEEE INFOCOM*, 2017.

[20] P. Sermpezis, T. Spyropoulos, L. Vigneri, and T. Giannakas, "Femto-caching with soft cache hits: Improving performance with related content recommendation," in *Proc. IEEE GLOBECOM*, 2017.

[21] C. Liang and F. R. Yu, "Wireless network virtualization: A survey, some research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 358–380, 2015.

[22] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. in Artif. Intell.*, pp. 4:2–4:2, 2009.

[23] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.

[24] S. Sesia, I. Toufik, and M. Baker, *LTE, The UMTS Long Term Evolution: From Theory to Practice*. Wiley Publishing, 2009.

[25] M. Dehghan, L. Massoulie, D. Towsley, D. Menasche, and Y. Tay, "A utility optimization approach to network cache design," in *Proc. IEEE INFOCOM*, 2016.

[26] A. Krause and D. Golovin, "Submodular function maximization," *Tractability: Practical Approaches to Hard Problems*, vol. 3, no. 19, p. 8, 2012.

[27] G. L. Nemhauser and L. A. Wolsey, "Best algorithms for approximating the maximum of a submodular set function," *Mathematics of operations research*, vol. 3, no. 3, pp. 177–188, 1978.

[28] M. Sviridenko, "A note on maximizing a submodular set function subject to a knapsack constraint," *Operations Research Letters*, vol. 32, no. 1, pp. 41–43, 2004.

[29] P. Sermpezis, T. Spyropoulos, L. Vigneri, and T. Giannakas, "Femto-caching with soft cache hits: Improving performance through recommendation and delivery of related content," *available at https://arxiv.org/abs/1702.04943*, 2017.

[30] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a monotone submodular function subject to a matroid constraint," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1740–1766, 2011.

[31] Y. Filmus and J. Ward, "Monotone submodular maximization over a matroid via non-oblivious local search," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 514–542, 2014.

[32] "Internet.org by Facebook." https://info.internet.org/, 2017.

[33] "free basics by Facebook." https://0.freebasics.com/desktop, 2017.

[34] http://netsg.cs.sfu.ca/youtubedata/, 2007.

[35] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proc. ACM SIGIR*, pp. 43–52, 2015.

[36] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, p. 19, 2016.

[37] K. Poularakis, G. Iosifidis, A. Argyriou, and L. Tassiulas, "Video delivery over heterogeneous cellular networks: Optimizing cost and performance," in *Proc. IEEE INFOCOM*, pp. 1078–1086, 2014.

[38] A. Ghosh, N. Mangalvedhe, R. Ratasuk, B. Mondal, M. Cudak, E. Visotsky, T. A. Thomas, J. G. Andrews, P. Xia, H. S. Jo, H. S. Dhillon, and T. D. Novlan, "Heterogeneous cellular networks: From theory to practice," *IEEE Comm. Magazine*, vol. 50, no. 6, pp. 54–64, 2012.

[39] J. G. Andrews, H. Claussen, M. Dohler, S. Rangan, and M. C. Reed, "Femtocells: Past, present, and future," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 497–508, 2012.

[40] J. G. Andrews, "Seven ways that hetnets are a cellular paradigm shift," *IEEE Communications Magazine*, vol. 51, no. 3, pp. 136–144, 2013.

[41] T. Spyropoulos and P. Sermpezis, "Soft cache hits and the impact of alternative content recommendations on mobile edge caching," in *Proc. ACM Workshop on Challenged Networks (CHANTS)*, pp. 51–56, 2016.

[42] W. C. Ao and K. Psounis, "Distributed caching and small cell cooperation for fast content delivery," in *Proc. ACM MobiHoc*, 2015.

[43] A. Liu and V. K. Lau, "Cache-enabled opportunistic cooperative mimo for video streaming in wireless systems," *IEEE Trans. Signal Processing*, 2015.

[44] D. Munaro, C. Delgado, and D. S. Menasché, "Content recommendation and service costs in swarming systems," in *Proc. IEEE ICC*, 2015.

[45] D. K. Krishnappa, M. Zink, and C. Griwodz, "What should you cache?: a global analysis on youtube related video caching," in *Procc. ACM NOSSDAV Workshop*, pp. 31–36, 2013.

[46] S. Khuller, A. Moss, and J. S. Naor, "The budgeted maximum coverage problem," *Information Processing Letters*, vol. 70, no. 1, pp. 39–45, 1999.

[47] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proc. ACM SIGKDD*, pp. 420–429, 2007.

Appendix A
Proof of Corollary 1

Sub-case 1. Since the exact per-user utilities are not known, we calculate the SCHR given in Eq. (1) by taking the conditional expectations on $F_{kn}(x)$. Denoting the corresponding pdf as $f_{kn}(x)$, we proceed as follows:

$$\begin{aligned}
SCHR &= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot E\left[\left(1-\prod_{n=1}^{K}\left(1-u_{kn}^i\cdot x_n\right)\right)\right] \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1-E\left[\prod_{n=1}^{K}\left(1-u_{kn}^i\cdot x_n\right)\right]\right) \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1-\prod_{n=1}^{K}E\left[\left(1-u_{kn}^i\cdot x_n\right)\right]\right) \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1-\prod_{n=1}^{K}\int\left(1-t\cdot x_n\right)\cdot f_{kn}(t)dt\right) \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1-\prod_{n=1}^{K}\left(1-\left(\int t\cdot f_{kn}(t)dt\right)\cdot x_n\right)\right) \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1-\prod_{n=1}^{K}\left(1-E[u_{kn}^i]\cdot x_n\right)\right)
\end{aligned}$$

where (i) the third equation holds since the utilities for different content pairs {k,n} are independent, and thus the expectation of their product is equal to the product of their expectations, and (ii) we denoted

$$E[u_{kn}^i] \equiv \int t\cdot f_{kn}(t)dt = \int(1-F_{kn}(t))dt$$

and the above equation holds since u_{kn}^i is a positive random variable.

Sub-case 2 follows straightforwardly.

Appendix B
Proof of Lemma 2

We prove here the NP-hardness of the optimal cache allocation for a single cache with soft cache hits. Let us consider an instance of Optimization Problem 1, where the utilities are equal among all users and can be either 1 or 0, i.e., $u_{kn}^i = u_{kn},\ \forall i \in \mathcal{N}$ and $u_{kn} \in \{0,1\},\ \forall k,n \in \mathcal{K}$. We denote as $\mathcal{R}_k$ the set of contents related to content k, i.e.

$$\mathcal{R}_k = \{n \in \mathcal{K} : n \neq k, u_{kn} > 0\} \text{ (related content set)} \quad (21)$$

Consider the content subsets $\mathcal{S}_k = \{k\} \cup \mathcal{R}_k$. Assume that only content k is stored in the cache ($x_k = 1$ and $x_n = 0, \forall n \neq k$). All requests for contents in S_k will be satisfied (i.e. "covered" by content k), and thus SCHR will be equal to $\sum_{i\in\mathcal{N}}\sum_{n\in S_k} p_n^i \cdot q_i$. When more than one contents are stored in the cache, let $\mathcal{S}^{'}$ denote the union of all contents covered by the stored ones, i.e., $\mathcal{S}^{'} = \bigcup_{\{k:x_k=1\}} S_k$. Then, the SCHR will be equal to $\sum_{i\in\mathcal{N}}\sum_{n\in S'} p_n^i \cdot q_i$. Hence, the Optimization Problem 1 becomes equivalent to

$$\max_{\mathcal{S}^{'}} \sum_{n\in\mathcal{S}^{'}} p_n^i \cdot q_i \qquad s.t. \quad |\{k : x_k = 1\}| \leq C.$$

This corresponds to the the *maximum coverage problem with weighted elements*, where "elements" (to be "covered") correspond to the contents $i \in \mathcal{K}$, weights correspond to the probability values $p_n^i \cdot q_i$, the number of selected subsets $\{k : x_k = 1\}$ must be less than C, and their union of covered elements is $\mathcal{S}^{'}$. This problem is known to be a NP-hard problem [46], and thus the more generic problem (with different u_{kn}^i and $0 \leq u_{kn} \leq 1$) is also NP-hard.

Appendix C
Proof of Lemma 3

The objective function of Eq. (5) $f(X) : \{0,1\}^K \to \mathbb{R}$ is equivalent to a set function $f(S) : 2^{\mathcal{K}} \to \mathbb{R}$, where $\mathcal{K}$ is the finite *ground set* of contents, and $S = \{k \in \mathcal{K} : x_k = 1\}$. In other words,

$$f(S) \equiv \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(1-\prod_{n\in S}\left(1-u_{kn}^i\right)\right). \quad (22)$$

A set function is characterised as *submodular* if and only if for every $A \subseteq B \subset V$ and $\ell \in V \backslash B$ it holds that

$$\left[f\left(A\cup\{\ell\}\right)-f\left(A\right)\right]-\left[f\left(B\cup\{\ell\}\right)-f\left(B\right)\right] \geq 0 \quad (23)$$

From Eq. (5), we first calculate

$$\begin{aligned}
f\left(A\cup\{\ell\}\right)-f\left(A\right) &= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i q_i \left(1-\prod_{n\in A\cup\{\ell\}}\left(1-u_{kn}^i\right)\right) \\
&\quad -\sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i q_i \left(1-\prod_{n\in A}\left(1-u_{kn}^i\right)\right) \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(\prod_{n\in A}\left(1-u_{kn}^i\right)-\prod_{n\in A\cup\{\ell\}}\left(1-u_{kn}^i\right)\right) \\
&= \sum_{i=1}^{N}\sum_{k=1}^{K} p_k^i \cdot q_i \cdot \left(u_{k\ell}^i \cdot \prod_{n\in A}\left(1-u_{kn}^i\right)\right).
\end{aligned}$$

Then,

$$
\begin{aligned}
&[f(A\cup\{\ell\})-f(A)]-[f(B\cup\{\ell\})-f(B)]=\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\left(u_{k\ell}^i\prod_{n\in A}(1-u_{kn}^i)\right)\\
&\qquad-\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\left(u_{k\ell}^i\prod_{n\in B}(1-u_{kn}^i)\right)\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\cdot u_{k\ell}^i\cdot\left(\prod_{n\in A}(1-u_{kn}^i)-\prod_{n\in B}(1-u_{kn}^i)\right)\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\cdot u_{k\ell}^i\cdot\prod_{n\in A}(1-u_{kn}^i)\cdot\left(1-\prod_{n\in B\setminus A}(1-u_{kn}^i)\right)
\end{aligned}
$$

The above expression is always ≥ 0, which proves the submodularity for function f.

Furthermore, the function f is characterised as *monotone* if and only if $f(B)\geq f(A)$ for every $A\subseteq B\subset V$. In our case, this property is shown as

$$
\begin{aligned}
f(B)-f(A)&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\cdot\left(1-\prod_{n\in B}(1-u_{kn}^i)\right)\\
&\qquad-\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\cdot\left(1-\prod_{n\in A}(1-u_{kn}^i)\right)\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\cdot\left(\prod_{n\in A}(1-u_{kn}^i)-\prod_{n\in B}(1-u_{kn}^i)\right)\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i q_i\cdot\prod_{n\in A}(1-u_{kn}^i)\cdot\left(1-\prod_{n\in B\setminus A}(1-u_{kn}^i)\right)\geq 0
\end{aligned}
$$

Appendix D
Proof of Theorem 2

Following similar arguments as in the proof of Lemma 2, the Optimization Problem 2 can be shown to be equivalent to the *budgeted maximum coverage problem with weighted elements*, which is an NP-hard problem [46].

In Algorithm 2, we first calculate a solution $S^{(1)}$ returned by a modified version (ModifiedGreedy) of the greedy algorithm (line 1). The differences between the greedy algorithm (e.g., Algorithm 1) and ModifiedGreedy, are that the latter: (a) each time selects to add in the cache the content that increases the most the fraction of the objective function over its own size (line 13), and (b) considers every content, until there is no content that can fit in the cache (lines 14-20). Then, Algorithm 2 calculates the solution $S^{(2)}$ that the greedy algorithm would return if all contents were of equal size (line 2). The returned solution, is the one between $S^{(1)}$ and $S^{(2)}$ that achieves a higher value of the objective function (lines 3-7).

Hence, Algorithm 2 is a "fast-greedy" type of approximation algorithm. Since, the objective function was shown to be submodular and monotone in Lemma 3, our fast greedy approximation algorithm can achieve a $\frac{1}{2}\cdot\left(1-\frac{1}{e}\right)$-approximation solution (in the worst case), when there is a Knapsack constraint, using similar arguments as in [47].

Appendix E
Proof of Lemma 5

Item (1): Optimization Problem 4 is of the exact same nature as Optimization Problem 3, so it follows that it is NP-hard.

Item (2): We proceed similarly to the proof of Lemma 3. The objective function of Eq. (17) $f(X):\{0,1\}^{K\times M}\rightarrow\mathbb{R}$ is equivalent to a set function $f(S):2^{\mathcal{K}\times\mathcal{M}}\rightarrow\mathbb{R}$, where $\mathcal{K}$ and $\mathcal{M}$ are the finite *ground sets* of contents and SCs, respectively, and $S=\{k\in\mathcal{K},j\in\mathcal{M}:x_{kj}=1\}$:

$$
f(S)\equiv\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i\cdot E_{G_i}\left[\max_{(n,j)\in S}\left(u_{kn}^i\cdot Q_{ij}\right)\right]\tag{24}
$$

For all sets $A\subseteq B\subset V$ and {content, SC} tuples $(\ell,m)\in V\setminus B$, we get

$$
\begin{aligned}
&f(A\cup\{(\ell,m)\})-f(A)=\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i\cdot E_{G_i}\left[\max_{(n,j)\in A\cup\{(\ell,m)\}}\left(u_{kn}^iQ_{ij}\right)\right]\\
&\qquad-\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i\cdot E_{G_i}\left[\max_{(n,j)\in A}\left(u_{kn}^iQ_{ij}\right)\right]\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i\cdot E_{G_i}\left[R\left(u_{k\ell}^i\cdot Q_{im}-\max_{(n,j)\in A}\left(u_{kn}^iQ_{ij}\right)\right)\right]
\end{aligned}
$$

where in the last equation we use the *ramp function* defined as $R(x)=x$ for $x\geq 0$ and $R(x)=0$ for $x<0$. Subsequently,

$$
\begin{aligned}
&[f(A\cup\{(\ell,m)\})-f(A)]-[f(B\cup\{(\ell,m)\})-f(B)]=\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i\cdot E_{G_i}\left[R\left(u_{k\ell}^iQ_{im}-\max_{(n,j)\in A}\left(u_{kn}^iQ_{ij}\right)\right)\right.\\
&\qquad\left.-R\left(u_{k\ell}^iQ_{im}-\max_{(n,j)\in B}\left(u_{kn}^iQ_{ij}\right)\right)\right]
\end{aligned}
$$

The above equation is always ≥ 0 (which proves that the objective function Eq. (17) is *submodular*), since the ramp function is monotonically increasing and comparing the two arguments of the function $R(x)$ in the above equation, gives

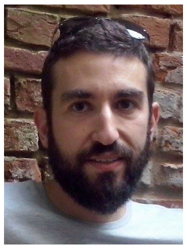
$$
\begin{aligned}
&u_{k\ell}^iQ_{im}-\max_{(n,j)\in A}\left(u_{kn}^iQ_{ij}\right)-\left(u_{k\ell}^iQ_{im}-\max_{(n,j)\in B}\left(u_{kn}^iQ_{ij}\right)\right)\\
&=\max_{(n,j)\in B}\left(u_{kn}^iQ_{ij}\right)-\max_{(n,j)\in A}\left(u_{kn}^iQ_{ij}\right)\geq 0
\end{aligned}
$$

since B is a superset of A and therefore its maximum will be at least equal or greater than the maximum value in set A.

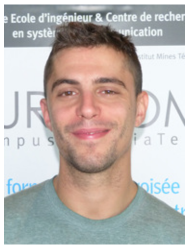
Similarly, since $A\subseteq B$ it holds

$$
\begin{aligned}
&f(B)-f(A)=\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}p_k^i\cdot E_{G_i}\left[\left(\max_{(n,j)\in B}\left(u_{kn}^iQ_{ij}\right)-\max_{(n,j)\in A}\left(u_{kn}^iQ_{ij}\right)\right)\right]\geq 0
\end{aligned}
$$

which proves that the Eq. (17) is *monotone*.



Pavlos Sermpezis received the Diploma in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Greece, and a PhD in Computer Science and Networks from EURECOM, Sophia Antipolis, France. He is currently a post-doctoral researcher at FORTH, Greece. His main research interests are in modeling and performance analysis for communication networks and protocols, and Internet routing.



Theodoros Giannakas received the Diploma in Electrical and Computer Engineering from the University of Patras, Greece, and his MSc in Wireless Communications from the University of Southampton, UK. He is currently pursuing his PhD in Computer Science and Networks at EURECOM, Sophia Antipolis, France. His main research includes mobile networks, caching, recommendation systems, and optimization.

Thrasyvoulos Spyropoulos received the Diploma in Electrical and Computer Engineering from the National Technical University of Athens, Greece, and a Ph.D degree in Electrical Engineering from the University of Southern California. He was a post-doctoral researcher at INRIA and then, a senior researcher with the Swiss Federal Institute of Technology (ETH) Zurich. He is currently an Assistant Professor at EURECOM, Sophia-Antipolis. He is the recipient of the best paper award in IEEE SECON 2008, and IEEE WoWMoM 2012.

Luigi Vigneri received the Diploma in Computer Engineering from Politecnico di Torino, Italy, and Telecom ParisTech, Paris, France, and a PhD in Computer Science and Networks from EURECOM, Sophia Antipolis, France. He is currently a post-doctoral researcher at Huawei Research Lab Paris, France. His main research interests are in online machine learning, data analysis, convex optimization, and mobile edge caching.