Approximation Algorithms for Mobile Data Caching in Small Cell Networks

Konstantinos Poularakis, George Iosifidis, and Leandros Tassiulas, *Fellow, IEEE*

***Abstract*—Small cells constitute a promising solution for managing the mobile data growth that has overwhelmed network operators. Local caching of popular content items at the small cell base stations (SBSs) has been proposed to decrease the costly transmissions from the macrocell base stations without requiring high capacity backhaul links for connecting the SBSs with the core network. However, the caching policy design is a challenging problem especially if one considers realistic parameters such as the bandwidth capacity constraints of the SBSs that can be reached in congested urban areas. We consider such a scenario and formulate the joint routing and caching problem aiming to maximize the fraction of content requests served locally by the deployed SBSs. This is an NP-hard problem and, hence, we cannot obtain an optimal solution. Thus, we present a novel reduction to a variant of the facility location problem, which allows us to exploit the rich literature of it, to establish algorithms with approximation guarantees for our problem. Although the reduction does not ensure tight enough bounds in general, extensive numerical results reveal a near-optimal performance that is even up to 38% better compared to conventional caching schemes using realistic system settings.**

***Index Terms*—Caching and routing algorithms, network optimization, facility location problems, small cell networks.**

I. Introduction

TO cope with the mobile data traffic explosion [2], [3] mobile network operators (MNOs) deploy small cell base stations (SBSs) which operate in conjunction with the macrocell base stations (MBSs) [4]. This architecture benefits both the MNO by replacing the long-range costly transmissions of the MBSs, and the users by offering them high-capacity, energy-prudent communication links. However, the operation of these small cells presumes the existence of high-speed backhaul links connecting the deployed base stations with the core network. This is currently one of the major operation cost components for MNOs [5] who seek methods for reducing it [6].

Manuscript received September 30, 2013; revised February 8, 2014 and June 24, 2014; accepted August 13, 2014. Date of publication August 26, 2014; date of current version October 17, 2014. Part of this work has been accepted to the proceedings of IEEE GLOBECOM, Atlanta, GA, USA, December 9–13, 2013. The associate editor coordinating the review of this paper and approving it for publication was C. W. Tan.

The authors are with the Department of Electrical and Computer Engineering, University of Thessaly, 38221 Volos, Greece and also with the Centre for Research and Technology Hellas (CERTH), 57001 Thessaloniki, Greece (e-mail: kopoular@inf.uth.gr; giosifid@inf.uth.gr; leandros@inf.uth.gr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCOMM.2014.2351796

A. Motivation

Decentralized caching architectures have been recently proposed [7]–[10] with the goal to minimize peak traffic—and subsequently, the cost—of these backhaul links. The main idea is to cache in advance (during off-peak demand) popular content items at the SBSs so as to reduce, especially during peak traffic hours, the requests that are routed over the backhaul links to the core network. Field trials [5] have revealed that this architecture can reduce the peak traffic on base stations by over 45% compared to schemes with no caching. Therefore, it is not surprising that related solutions have already attracted the interest of industry [11]–[13].

Given the vast set of the content items, the challenge in this context is to find the optimal caching policy. That is, decide which items should be cached at each base station, so as to maximize the portion of user requests that are satisfied locally by the SBSs, without using backhaul links or employing the MBSs. Unfortunately though, this has been proved to be in general an NP-hard problem [7].

Deriving the optimal caching policy becomes even more challenging if one considers *massive* content delivery scenarios, e.g., in populated areas or during peak traffic hours. In these cases, mobile data delivery will be constrained by the transmission capacity of the SBSs. Obviously, to deliver a content item to a user, it does not suffice to have it cached at a base station within the user's transmission range, but additionally the base station should have enough capacity to deliver it. Prior works assume that the transmission capacity is rarely the bottleneck for the caching base stations. Clearly, this is not a realistic assumption for dense urban areas where user content demand is often massive.

B. Methodology and Contributions

In this work, we consider the scenario of massive content delivery through SBSs with *hard bandwidth constraints* that bottleneck the data transmission to mobile users. We explain through simple examples that, in this case, the MNO's caching policy has to take into account the bandwidth capacity constraints. Additionally, user requests should not be trivially routed to any base station that has cached the content items of interest. Instead, the operator must explicitly devise the routing policy, i.e., determine which requests to route to each SBS. This means that the routing and caching policies must be jointly designed. To achieve this, we provide an analytical framework for this *joint routing and caching for un-splittable requests* (JRC-UR) problem, the solution of which maximizes the content requests that are satisfied by the SBSs.

This problem is very important and equally challenging to solve. Namely, the already NP-hard caching problem is further compounded due to the additional bandwidth constraints of the SBSs. Besides, our model additionally takes into account (i) the *heterogeneity of the base stations* which may have different cache sizes and transmission capacities, and (ii) the *variation of request patterns of the users* which may ask for different content items with different intensity.

We group users into *user classes* based on their geographic location, and allow a user class to generate more than one requests for each file. We consider the case that user *requests are unsplittable*, which implies that each one of the requests is entirely satisfied by one base station, i.e., a user that requests a file will not receive different *parts* of it from different base stations, but a complete replica from a single base station.[1] This is of major importance, since associating a user request with multiple base stations incurs the extra effort to synchronize the communication. Besides, user association cannot change in a very small time scale as base station reselection requires a time interval of several seconds [14]. Hence, the unsplittable case is the most realistic and challenging scenario in caching problems, and a constraint that is often relaxed to simplify analysis [15], [16].

To address these issues we propose a novel mapping of the JRC-UR problem to a variant of the facility location problem known as the *Unsplittable Hard-Capacitated Metric Facility Location Problem* (UHCMFL). The UHCMFL problem has been studied extensively, and there exist a variety of bi-criteria approximation algorithms for its solution [17]–[22]. A bi-criteria (α, β)-approximation algorithm ensures an α-approximation solution under the assumption that the facility capacities can be violated up to β times. We prove that the JRC-UR can be reduced in polynomial time to UHCMFL. Moreover, we provide a methodology for transforming the above bi-criteria algorithms to standard approximation algorithms that do not require violation of capacities. This is very important as, in many settings, the base stations (which correspond to facilities) have hard and not easy-to-expand capacity constraints [23].[2]

The presented mapping leads to an optimization framework for small cell caching problems where we can employ, after proper modifications, the algorithms that have been derived for the class of UHCMFL problems. In general, the theoretical approximation bounds for the JRC-UR problem are highly sensitive to variation in the number of content items and the network load, and consequently are not always particularly tight. Even so, to the best of the authors knowledge, this is the first work that provides a solution with provable approximation guarantees for the joint content placement and request routing problem in small cell networks. More importantly, based on extensive numerical study, we show that the casted UHCMFL algorithms, that are applicable due to the proposed transformation, perform very well and close to the optimal solution. Additionally, we show numerically that the obtained solution outperforms conventional heuristic caching algorithms (with static, non-optimized routing).

The main technical contributions of this work can be summarized as follows:

- *Modeling*. We formulate the JRC-UR problem which considers important features such as the resource heterogeneity (storage and bandwidth capacities) of the small cell base stations, and the different content request patterns of the users. Its solution yields the *joint routing and caching policy* that maximizes the requests served by the SBSs.
- *Connection to Facility Location Problem*. We reduce the JRC-UR problem to the *Unsplittable Hard-Capacitated Metric Facility Location Problem* (UHCMFL) [21]. This reduction reveals the potential for exploiting the rich literature in facility location problems in order to optimize caching in small cell networks.
- *Facility-location Inspired Algorithms*. We present an approximation framework based on the aforementioned reduction. Particularly, we show that every approximation algorithm for UHCMFL can be transformed to a respective algorithm with an approximation ratio for the JRC-UR problem. These algorithms perform in practice very well, and far better than the worst-case conditions indicate.
- *Performance Evaluation*. We evaluate numerically one of the derived approximation algorithms in representative scenarios, and find that its performance (in terms of requests routed to MBS) is less than 10% worse than the optimal solution. This result is further improved as the cache sizes, the steepness of the file popularity distribution, and/or the total request load increases. Moreover, we show that the proposed scheme outperforms typical greedy algorithms up to 38%.

The rest of the paper is organized as follows. Section II describes the system model and the assumptions, and introduces formally the problem. In Section III we reduce the problem to the UHCMFL problem. Section IV presents an approximation framework based on the above reduction, whereas Section V provides the numerical results. In Section VI we review our contribution compared to related works, and we conclude in Section VII.

II. System Model and Problem Formulation

In this section, we first introduce the system model and explain the considered network architecture. In the sequel, we provide a simple, yet representative, example that motivates the joint design of routing and caching policies, and we formally introduce the respective optimization problem.

A. System Model

We consider a single macrocell[3] in which the mobile network operator (MNO) serves the content requests submitted by a set $\mathcal{K} = \{1, 2, \ldots, K\}$ of $K = |\mathcal{K}|$ classes of mobile users (MUs). Each class represents the users lying in a certain geographic

[1] However, please notice that requests for the same file generated by users in the same class can be satisfied by different base stations.

[2] Small-cells could obtain extra bandwidth in various ways, for instance by spectrum reuse or from Cognitive Mobile Virtual Network Operator.

[3] The study can be directly extended for more macrocells.

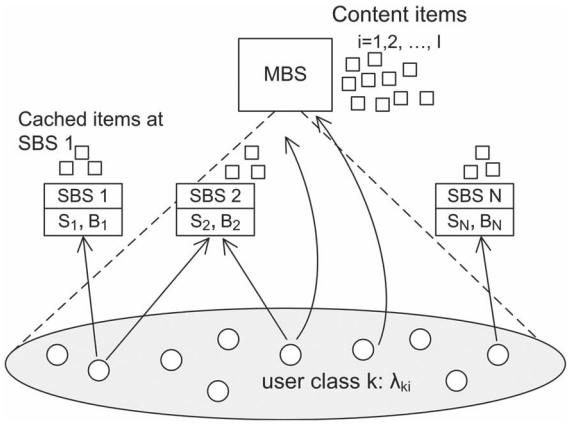

Fig. 1. Mobile users are randomly distributed in the coverage regions of the SBSs. Each SBS n has certain storage and bandwidth capacity of S_n and B_n units respectively.

area.[4] Hence, it is possible to have more than one requests for one file originating from the same point. Also, there exists a set $\mathcal{N} = \{1, 2, \ldots, N\}$ of $N = |\mathcal{N}|$ small cell base stations (SBSs) which operate in conjunction with the macrocell base station (MBS), yielding a heterogeneous cellular network. This two-layer architecture is depicted in Fig. 1.

We consider the case that the SBSs operate in disjoint subchannels than the MBS [24], [25]. Also, we assume that neighboring SBSs are assigned orthogonal frequency bands and/or employ enhanced inter-cell interference coordination techniques (eICIC) proposed in LTE Rel. 10 [26]. We assume that time is slotted and we study the system for one time period T. Each SBS $n \in \mathcal{N}$ has a certain transmission capacity, i.e., it can deliver $B_n \geq 0$ data bytes within period T. Additionally, each SBS $n \in \mathcal{N}$ is endowed with a storage capacity of $S_n \geq 0$ bytes.

Let the set $\mathcal{I}$ indicate a static collection of $I = |\mathcal{I}|$ content items (or, files). For notational convenience, we assume that all files have the same size s. This assumption can be easily removed as, in real systems, files can be divided into blocks of the same length or by leveraging advanced coding techniques [7], [15]. We denote with $\lambda_{ki} \in \mathcal{Z}^+$ the expected number of requests for file $i \in \mathcal{I}$ generated by user class $k \in \mathcal{K}$ within T. Observe that a user class consists of many users and thus can generate more than one requests for the same file. User requests may change over consecutive time periods but are considered fixed (and known) within each period. This is a realistic assumption for proactive caching as the popularity distribution of the files changes slowly [7], [27], [28]. Similar approaches have been followed for modeling user demand in cellular networks [29]. The coverage areas of the SBSs are *overlapping*. Let $\mathcal{N}_k \subseteq \mathcal{N}$ denote the set of SBSs that are in communication range with user class k. Then, a request generated by k can be satisfied by any of the SBSs in $\mathcal{N}_k$ that owns a copy of the requested item. The requests that cannot be satisfied locally, i.e., by any SBS, are routed to the MBS. Our goal is to minimize this latter quantity which depends both on the caching and the routing policy of the operator.

[4]Hereafter, we may use the term *user* $k \in \mathcal{K}$ to refer to *user class* $k \in \mathcal{K}$.

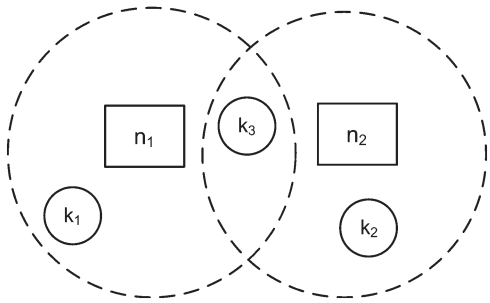

Fig. 2. An example with 2 SBSs (n_1 and n_2) and 3 users (k_1, k_2, k_3). The circles denote the transmission range of each SBS.

B. Motivating Example

Consider the system depicted in Fig. 2 with two SBSs (n_1 and n_2) and three users (k_1, k_2 and k_3). The circles represent the coverage areas of the SBSs while all the users are also covered by the MBS (not shown). There are also two equal-sized files denoted i_1 and i_2. Each SBS can cache at most one file due to its limited storage capacity. Also, because of the bandwidth limitations, n_1 can serve at most 5 requests and n_2 can serve at most 10 requests. User class k_1 requests file i_1 1 time, k_2 requests i_1 2 times and k_3 requests file i_2 10 times. The optimal routing and caching strategy is the one that maximizes the requests satisfied by n_1 and n_2. In this example, this policy dictates to cache i_1 to n_1, and i_2 to n_2. Then, n_1 serves the request for i_1 generated by k_1, and n_2 serves all the requests for i_2 generated by k_3. Hence, only 2 requests need to be served by the MBS.

However, if we omit the transmission capacity constraints, then the *optimal caching policy changes*: it places i_2 to n_1, and i_1 to n_2. Then, n_1 handles all the requests of k_3, and n_2 handles all the requests of k_2, letting only the one request generated by k_1 to be routed to the MBS. Nevertheless, in practice, n_1 will serve only half of incoming requests (due to limited capacity) and redirect the rest to the MBS. Hence, the MBS will have to serve $6 > 2$ requests.

This example demonstrated that *ignoring the SBSs' bandwidth capacities when designing the caching policy leads the system to inefficient operating points*, for the case of massive content requests where the capacity limits of the SBSs are reached. In the sequel, we formalize the respective problem for the general case.

C. Problem Formulation

Let us introduce the binary decision variable $x_{ni} \in \{0, 1\}$ which indicates whether file $i \in \mathcal{I}$ is placed at the cache of SBS $n \in \mathcal{N}$ or not. We also define the respective *caching policy* matrix:

$$\boldsymbol{x} = (x_{ni} : n \in \mathcal{N},\, i \in \mathcal{I}). \tag{1}$$

Additionally, let the integer decision variable $y_{ni}^k \in \mathcal{Z}^+$ indicate the number of requests for file i generated by user class k that are routed to SBS n. Also, $y_{Mi}^k \in \mathcal{Z}^+$ denotes the number of requests for file i generated by user $k \in \mathcal{K}$ that are routed to

the MBS which is denoted with M. The *routing policy* of the operator is described by the following matrix:

$$\boldsymbol{y} = \left(y_{ni}^k : n \in \mathcal{N} \cup \{M\}, i \in \mathcal{I}, k \in \mathcal{K}\right). \quad (2)$$

Observe that each one of the λ_{ki} requests for file i generated by user class k must be satisfied by exactly one SBS, $\forall i \in \mathcal{I}, k \in \mathcal{K}$. This means that each variable y_{ni}^k is allowed to take value in the *integer set* $\{0, 1, \ldots, \lambda_{ki}\}$ ("un-splittable requests"). The above integrality constraint makes the problem even harder compared to the simplified case that any fraction of the total user demand for a file is allowed to be routed to multiple SBSs, i.e., the variable y_{ni}^k take values in the *real set* $[0, \lambda_{ki}]$ ("splittable requests") [15], [16].

As we explained in the previous example, the routing policy should take into account the bandwidth capacity constraint B_n of each SBS $n \in \mathcal{N}$. Clearly, if an SBS is already congested, it cannot serve additional user requests. Moreover, routing decisions are coupled with the respective caching decisions: a request is routed to an SBS only if the latter has the requested content item cached. At the same time, the caching policy must respect the storage capacity S_n of each base station $n \in \mathcal{N}$.

Summarizing, the problem of devising the joint routing and caching policy for un-splittable requests (JRC-UR problem) which minimizes the requests routed to the MBS, can be formulated as follows:

$$\min_{\boldsymbol{x},\boldsymbol{y}} \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{I}} y_{Mi}^k \quad (3)$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{I}} x_{ni} s \leq S_n, \ \forall n \in \mathcal{N}, \quad (4)$$

$$\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{I}} y_{ni}^k s \leq B_n, \ \forall n \in \mathcal{N}, \quad (5)$$

$$y_{ni}^k \leq x_{ni}\lambda_{ki}, \ \forall i \in \mathcal{I}, k \in \mathcal{K}, n \in \mathcal{N}, \quad (6)$$

$$y_{ni}^k = 0, \ \forall i \in \mathcal{I}, \ k \in \mathcal{K}, n \in \mathcal{N} \setminus \mathcal{N}_k, \quad (7)$$

$$\sum_{n\in\mathcal{N}\cup\{M\}} y_{ni}^k = \lambda_{ki}, \ \forall i \in \mathcal{I}, k \in \mathcal{K}, \quad (8)$$

$$x_{ni} \in \{0, 1\}, \ \forall n \in \mathcal{N}, \ i \in \mathcal{I}, \quad (9)$$

$$y_{ni}^k \in \mathcal{Z}^+, \ \forall n \in \mathcal{N} \cup \{M\}, i \in \mathcal{I}, \ k \in \mathcal{K}, \quad (10)$$

where inequalities (6) indicate that SBSs can not serve requests for files that are not in their caches. Constraints (7) denote that SBSs can not serve requests generated by users located out of their coverage areas, and (8) dictate that the system must serve all the requests (inelastic demand).[5] Finally, (9), (10) reveal the discrete nature of the optimization variables.

Clearly, the above problem is very hard to solve optimally. Namely, the following lemma holds.

Lemma 1: The JRC-UR problem is NP-hard.

Proof: The JRC-UR problem is a generalization of the *Helper Decision Problem* (HDP), described in [7], by incorporating the hard bandwidth constraints of the SBSs. Hence, problem HDP, which is NP-hard [7], can be directly reduced in polynomial time to our problem. Consequently, JRC-UR is also NP-hard. ■

[5]Notice that we have not included a capacity constraint for the MBS, assuming that it can accommodate all the unsatisfied requests even if this entails a very high OpEx cost.

TABLE I
KEY NOTATIONS

Symbol	Physical Meaning
k	Users belonging to user class $k \in \mathcal{K}$
S_n	Storage capacity of SBS n
B_n	Bandwidth capacity of SBS n
λ_{ki}	Number of requests of user class k for file i
x_{ni}	Caching decision for file i to SBS n
y_{ni}^k	Number of requests of user class k satisfied through SBS n
y_{Mi}^k	Number of requests of user class k satisfied through MBS
s	Size (bytes) of each content item
$\mathcal{N}_k$	Set of base stations in range with user class k

Finally, we summarize the key notation used throughout the paper in Table I.

III. REDUCTION TO A VARIANT OF THE FACILITY LOCATION PROBLEM

In this section our goal is to devise a polynomial time reduction of the JRC-UR problem to a well known facility location problem. This will help us in the sequel to derive approximation algorithms for our problem, by using the ones that have been designed for the facility location problem. Note that although, in general, reduction preserves only optimality (and not approximation bounds) [30], for our case it also holds that we can compute how much in the worst case the approximation ratio deteriorates, as we show in the next section. Hence, the reduction serves as the main building block for our optimization framework. Subsequently, we describe a polynomial time reduction of the JRC-UR problem to the following variant of the facility location problem [21]:

Definition 1—Unsplittable Hard-Capacitated Metric Facility Location Problem (UHCMFL): We are given a set $\mathcal{V}$ of $|\mathcal{V}|$ locations, where there is a subset $\mathcal{A} \subseteq \mathcal{V}$ of facilities, and a subset $\mathcal{B} \subseteq \mathcal{V}$ of clients. Let $d_i \geq 0$ denote the demand of client $i \in \mathcal{B}$. Besides, let $f_j \geq 0$ and $C_j \geq 0$ denote the opening cost and the capacity of facility $j \in \mathcal{A}$, respectively. Each client needs to assign its entire demand to a single open facility (*unsplittable*). Capacity C_j limits the total sum of demands served by facility j (*hard capacitated*). We denote by $c_{ij} \geq 0$ the unit cost incurred when serving one unit of demand of client i by facility j. We assume that these costs form a *metric*, i.e., they are non-negative, symmetric $(c_{ij} = c_{ji})$, and satisfy the triangle inequality: $c_{ij} + c_{jk} \geq c_{ik}, \ \forall i, j, k \in \mathcal{V}$.

The problem is to determine which subset of facilities $\mathcal{A}^* \subseteq \mathcal{A}$ should open, and which clients each one of them should serve (denoted by a function $\pi : \mathcal{B} \to \mathcal{A}^*$), so as to minimize the aggregate facility opening and servicing cost Q:

$$Q = \sum_{j\in A^*} f_j + \sum_{i\in\mathcal{B}} d_i c_{i\,\pi(i)}. \quad (11)$$

At the same time satisfying the capacity constraints $\sum_{\{i\in\mathcal{B}:\pi(i)=j\}} d_i \leq C_j, \ \forall j \in \mathcal{A}$.

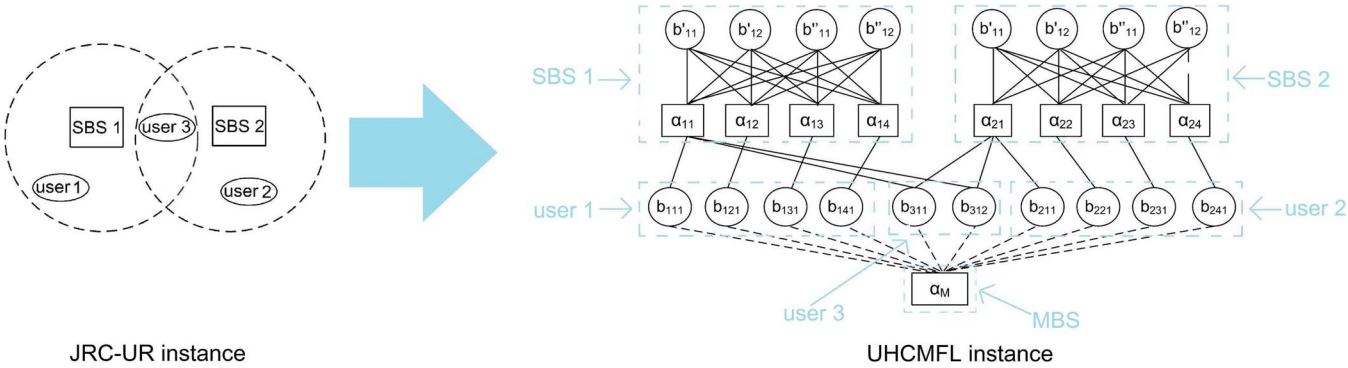

Fig. 3. An example of the reduction to the UHCMFL problem. We consider a setting with 1 MBS, 2 SBSs, and 3 users as shown on the left. The system parameters are $|\mathcal{I}| = 4$, $s = 1$, $S_1 = S_2 = 2$, and $B_1 = B_2 = 2$.

The connection between the UHCMFL and the JRC-UR problem is non-trivial. In fact, previous works in the literature that established reductions of caching problems to facility location problems focused on the simplified case that only *a single piece of content* is to be placed in the caches [21]. Our model substantially differs from these works, as it considers the practical case that *multiple files* exist, while the cache size and the bandwidth capacity of the SBSs are limited. To the best of our knowledge *this is the first work that shows that such a connection exists*. Theorem 1 describes this result.

Theorem 1: The JRC-UR problem is polynomial-time reducible to the UHCMFL problem.

We describe in detail this reduction and prove its validity in the following two subsections.

A. The Reduction

In this subsection, we analytically describe the reduction mentioned in Theorem 1. Particularly, we reduce any instance of the JRC-UR problem to an instance of the UHCMFL problem. Let F_{JRC-UR} be that instance of the UHCMFL problem. Then, F_{JRC-UR} is constructed as follows:

The set of facilities $\mathcal{A}$ consists of: (i) one facility named a_M for the MBS and (ii) a facility named a_{ni} for every SBS $n \in \mathcal{N}$ and every file $i \in \mathcal{I}$. The set of clients $\mathcal{B}$ comprises the following subsets: (i) $\mathcal{B}_1$ that contains λ_{ki} clients, $\forall k \in \mathcal{K}$ and $\forall i \in \mathcal{I}$, denoted as $b_{ki1}, b_{ki2} \ldots, b_{ki\lambda_{ki}}$, (ii) $\mathcal{B}_2$, with $|\mathcal{I}| - \lfloor S_n/s \rfloor$ clients, denoted b'_{n1}, b'_{n2} etc, $\forall n \in \mathcal{N}$, and (iii) subset $\mathcal{B}_3$ which contains $(\lfloor S_n/s \rfloor - 1)\lfloor B_n/s \rfloor$ clients, which are denoted b''_{n1}, b''_{n2} etc, $\forall n \in \mathcal{N}$. The symbol $\lfloor . \rfloor$ denotes rounding to the next lower integer. The capacity of the facility a_M is set to $+\infty$ and to B_n/s for each a_{ni}, $\forall n \in \mathcal{N}$, $i \in \mathcal{I}$. The demand of each client $b'_{ni} \in \mathcal{B}_2$ is equal to B_n/s. Each of the remaining clients $b_{kij} \in \mathcal{B}_1$ and $b''_{ni} \in \mathcal{B}_3$ has demand equal to 1.

Let c be an arbitrarily small positive constant. Then, the unit serving cost for each pair of a facility and a client is specified as follows: (1) Each pair of the form (a_M, b_{kij}), $\forall k, i, j$, has cost equal to $1 + 0.5 + c$. (2) Each pair of the form (a_{ni}, b_{kij}), such that $n \in \mathcal{N}_k$ and $j \in \{1, \ldots, \lambda_{ki}\}$, has cost equal to $0.5 + c$. (3) Each pair of the form (a_{ni}, b'_{nj}), $\forall n, i, j$, has cost equal to $0.5 + c$. (4) Each pair of the form (a_{ni}, b''_{nj}), $\forall n, i, j$, has cost equal to $0.5 + c$. The cost value of each of the remaining pairs is equal to the cost of the shortest path that unite this pair. Thus, the costs form a metric. Finally, the facility opening cost is set to zero for every facility.

Roughly speaking, the facility a_M represents the MBS and the facilities a_{ni}, $\forall i$, the SBS n. Hence, the facility capacity choices indicate that the MBS can serve all the user requests, while each SBS n can serve up to a limited number of requests. Each one of the clients of the type $b_{kij} \in \mathcal{B}_1$, $\forall k, i, j$ (whose demand is equal to one) represents one user request, while $b'_{ni} \in \mathcal{B}_2$, and $b''_{ni} \in \mathcal{B}_3$, $\forall n, i$ denote *virtual* user requests that are necessary to preserve the cache capacity and bandwidth constraints of the SBSs, as it will become clear in the following subsection.

Each solution for the F_{JRC-UR} problem can be mapped to a solution for the JRC-UR problem as follows:

- Rule 1: For each facility a_{ni} *not* serving any client of the form $b'_{nj} \in \mathcal{B}_2$, $\forall j$, place file i to the cache of SBS n.
- Rule 2: For each facility of the form a_{ni} serving a client of the form $b_{kij} \in \mathcal{B}_1$, $\forall n, i, k, j$ route the jth request of user k for file i to SBS n.
- Rule 3: The remaining requests are routed to the MBS.

Fig. 3 depicts the reduction for the example of Fig. 2. Here, we set the system values as follows: $|\mathcal{I}| = 4$, $s = 1$, $S_1 = S_2 = 2$ and $B_1 = B_2 = 2$. Each of the two first users requests every file once. User 3 performs two requests for the first file. Squares represent the facilities and circles the clients. Solid lines unite clients to facilities with cost $0.5 + c$. Dashed lines mean that the corresponding cost is $1 + 0.5 + c$. The cost value of each of the remaining pairs is equal to the cost of the shortest path that unites this pair. For example, the cost between the client b'_{11} and facility a_{21} is $1.5 + 3c$. The demand of each client is 1, except for the clients named as $b'_{ni} \in \mathcal{B}_2$, $\forall n, i$, whose demand is 2. The capacity of each facility is 2, except for the a_M facility, whose capacity is $+\infty$.

To help the reader understand the rationality behind the reduction, we also present a partition of the UHCMFL components, specified by the dashed rectangles and the arrowed labels with cyan color. Recall that the role of the clients in $\mathcal{B}_2$ and $\mathcal{B}_3$ is to preserve the cache space and bandwidth limitations of the SBSs, which are mapped to the facilities a_{ni}, $\forall n, i$. Hence, a logical partition of the UHCMFL components should include

the above into the same group. To this end, each SBS in our example corresponds to the 8 components in the top of the UHCMFL instance (namely 4 facilities and 4 clients), each user k to $\sum_{i\in\mathcal{I}}\lambda_{ki}$ of the bottom clients, and the MBS to the facility named a_M.

B. The Reduction Proof

We now prove that the preceding reduction holds, by proving the next two lemmas. Let D denote the total demand of the clients in F_{JRC-UR}. Then, we have:

Lemma 2: For every feasible solution of the JRC-UR problem with value C, there is a feasible solution to F_{JRC-UR} with total cost $C + D(c + 0.5)$.

Proof: We construct the solution to F_{JRC-UR} as follows:

(1) We open all the facilities at zero cost.
(2) For each file i *not* cached at SBS n, we assign the (entire) demand of one client of type $b'_{nj} \in \mathcal{B}_2,\ j \in \{1,\ldots,|\mathcal{I}| - \lfloor S_n/s\rfloor\}$ to the facility a_{ni}.
(3) For each request generated by a user k for a file i served by an SBS n, we assign the demand of one client of type $b_{kij} \in \mathcal{B}_1,\ j \in \{1,\ldots,\lambda_{ki}\}$ to the facility a_{ni}.
(4) The demand of a client of type $b''_{nj} \in \mathcal{B}_3,\ j \in \{1,\ldots,(\lfloor S_n/s\rfloor - 1)(\lfloor B_n/s\rfloor)\}$, is randomly assigned to one of the facilities of the form $a_{ni},\ \forall i \in \mathcal{I}$, without violating their capacity constraints.
(5) For each client $b_{kij} \in \mathcal{B}_1$ that has not been covered yet, we assign its demand to the facility a_M. Thus, every unit of demand of the clients was assigned to a facility. An assignment to the facility a_M incurs a per unit cost equal to $1 + 0.5 + c$, while all the other assignments incur a per unit cost equal to $0.5 + c$. By construction of the graph, the total demand assigned to a_M is equal to the number of requests that are routed to the MBS (C). Thus, the solution has cost equal to $C + D(0.5 + c)$. ■

Lemma 3: For every minimum cost solution of the F_{JRC-UR} instance with total cost C, there is a feasible solution to the JRC-UR problem with value $C - D(0.5 + c)$.

Proof: We construct the solution to JRC-UR problem as follows:

(1) *For each facility* a_{ni} not *serving any client of the form* $b'_{nj} \in \mathcal{B}_2,\ \forall j$, *place file* i *to the cache of SBS* n (Rule 1). Observe that each client $b'_{nj} \in \mathcal{B}_2,\ \forall j$, must be assigned to a facility of the form $a_{ni},\ \forall i \in \mathcal{I}$, at per unit cost $0.5 + c$. This is because each of the other choices incurs at least $1 + 0.5 + 3c$ per unit cost. Thus, the extra cost paid is at least $1 + 2c$. On the other hand, each client $b_{kij} \in \mathcal{B}_1,\ \forall k, i, j$, can always be assigned to the facility a_M at per unit cost $1 + 0.5 + c$. This means that the potential gain for assigning it to a facility of the form $a_{ni},\ \forall i \in \mathcal{I}$, at cost $0.5 + c$, is equal to 1, which is strictly lower than the extra cost paid above. We also observe that, the demand of each of these clients is equal to the capacity of each of the facilities of the form $a_{ni},\ \forall i \in \mathcal{I}$. There are $|\mathcal{I}| - \lfloor S_n/s\rfloor$ such clients. Thus, these clients fully occupy the capacity of $|\mathcal{I}| - \lfloor S_n/s\rfloor$ of these facilities. Consequently, exactly $\lfloor S_n/s\rfloor$ of the above facilities will remain *uncovered* corresponding to the files placed at the cache of SBS n.
(2) *For each facility of the form* a_{ni} *serving a client of the form* $b_{kij} \in \mathcal{B}_1,\ \forall n, i, k, j$ *route the* jth *request of user* k *for file* i *to SBS* n (Rule 2). Observe that each of the clients of type $b''_{nj} \in \mathcal{B}_3,\ \forall j$, must be assigned to one of the $\lfloor S_n/s\rfloor$ *uncovered* facilities of the form $a_{ni},\ \forall i$, similarly to the above case. The capacity of each of these facilities is equal to B_n/s. There exist $(\lfloor S_n/s\rfloor - 1)\lfloor B_n/s\rfloor$ such clients, each of them with demand equal to 1. Thus, the remaining capacity suffices for serving at most B_n/s units of demand of the clients $b_{kij} \in \mathcal{B}_1,\ \forall k, i, j$. By construction, a client $b_{kij} \in \mathcal{B}_1$ can be served by a facility a_{ni} with cost equal to $0.5+c$ iff $n \in \mathcal{N}_k$. The cost for serving $b_{kij} \in \mathcal{B}_1$ by $a_{ni},\ \forall n \notin \mathcal{N}_k$ is more than the serving cost by a_M. Thus, at most B_n/s requests generated by users in the coverage area of an SBS n will be routed to $n,\ \forall n \in \mathcal{N}$. The remaining $C - D(0.5+c)$ requests will be routed to the MBS (Rule 3). ■

To avoid confusion, we need to emphasize that the key point of the reduction is to force all the clients $b'_{nj} \in \mathcal{B}_2$ and $b''_{nj} \in \mathcal{B}_3,\ \forall j$ to assign their (entire) demand on one of the facilities of the form $a_{ni},\ \forall i$, for each SBS $n \in \mathcal{N}$. To ensure this, we picked the servicing cost values appropriately. A different choice would be to set the cost between each one of the clients in the set $\mathcal{B}_2 \cup \mathcal{B}_3$ and one of the above facilities equal to zero, and to $+\infty$ for all the other choices. Then, we could set the cost for assigning clients in $\mathcal{B}_1$ from $0.5 + c$ to zero, from $1 + 0.5 + c$ to 1 and to $+\infty$ for the rest choices. Note that, although this new instance of the facility location problem would be equivalent to the JRC-UR problem, the costs would not form a metric any more. The non-metric version of the UHCMFL problem is much harder, and there are not any approximation algorithms for it in the literature. This is the reason that we added the quantity $0.5 + c$ to the cost value of each one of the above links and restricted the cost of each one of the rest links to be equal to the aggregate cost in the shortest path that unites the endpoint vertices.

Note that the reduction does not hold for $c \leq 0$, since the extra cost paid for assigning a client $b'_{nj} \in \mathcal{B}_2$ or $b''_{nj} \in \mathcal{B}_3$ to a facility $a_{n'i},\ n' \neq n$ can be lower than the potential gains, as explained in Lemma 3 (1).

IV. Approximation Algorithms

In this section, we present an approximation framework for the JRC-UR problem based on the reduction described in Section III. We first discuss existing approximation algorithms for the UHCMFL problem. Accordingly, we describe methods to extend them so as to tackle the problem under consideration. Additionally, we derive improved approximation ratios for the special case of equal transmission capacity SBSs. This is an important case because, more often than not, SBSs will be of the same type and hence they will have equal transmission capacities. Finally, elaborating on the derived approximation ratios, we show that these ratios are highly sensitive to variation in the number of content items and the network load, and consequently are not always particularly tight.

TABLE II
BI-CRITERIA BOUNDS FOR THE UHCMFL PROBLEM

Algorithm	Case	Bound	Complexity	Reference
1	Uniform	$(9, 4)$	P.	[17]
2	Uniform	$(7.62, 4.29)$	P.	[17]
3	Uniform	$(O(1), 2)$	P.	[18]
4	General	$(11, 2)$	P.	[22],[19],[17]
5	Uniform	$(5, 2)$	P.	[22],[20],[17]
6	Uniform	$(\log \lvert\mathcal{V}\rvert, 1+\epsilon)$	P.	[21]
7	General	$(\log \lvert\mathcal{V}\rvert, 1+\epsilon)$	QP.	[21]
8	Uniform	$(10.173, 1.5)$	P.	[22]
9	Uniform	$(30.432, \frac{4}{3})$	P.	[22]

A. Approximation Ratios for the UHCMFL Problem

It is NP-hard even to approximate the solution of UHCMFL problem. Hence, previous works [17]–[22] focused on obtaining *bi-criteria* approximation algorithms. Formally, an (α, β)-bi-criteria approximation algorithm finds an infeasible solution with a cost at most $\alpha \geq 1$ times the optimal cost and aggregate demand assigned to each facility at most $\beta \geq 1$ times its capacity. Similarly, we can define an (α, β)-bi-criteria approximation algorithm for the JRC-UR problem, such that its solution violates the bandwidth capacities of the SBSs by at most a factor of β. Clearly, when β equals to one, a feasible solution is attained. We call the corresponding algorithm simply as an α-approximation algorithm.

Table II summarizes the existing results in the literature in chronological order. Notice that different results are obtained for the case that facilities have equal capacities (uniform case). Parameter $\epsilon > 0$ is arbitrarily small and $|\mathcal{V}|$ is the size of the facility location instance. A Quasi-polynomial time algorithm $(QP.)$ runs slower than polynomial time $(P.)$, yet faster than exponential time [21]. For example, the complexity of Algorithm 7 is $|\mathcal{V}|^{O(\log |\mathcal{V}|)}$.

Shmoys *et al.* [17] provided the first approximation algorithm for the UHCMFL problem (Algorithm 1). They used the filtering and rounding technique of Lin and Vitter to solve the splittable version of the problem, and then round the obtained solution to provide a (9, 4)-approximation algorithm for the unsplittable case. The first step requires solving the linear programming relaxation of the UHCMFL problem, and then rounding it to obtain a $g - close$ integer solution, i.e., a solution that assigns clients to facilities with cost at most g. The second step expresses the problem as one of assigning jobs to machines. Then, it rounds the current solution to a new one by solving an appropriately constructed instance of the maximum weight matching in a bipartite graph problem. The running time of this procedure is cubic to the size of the facility location instance. Authors in [17] also proposed a randomized variant of the above technique that provides an improved approximation guarantee (Algorithm 2). As it is well discussed in [22], applying the same rounding technique of [17] to the results for the splittable case provided in [18]–[20] yields even tighter approximation ratios for the unsplittable variant of the facility location problem (Algorithms 3–5).

The work in [21] provided an $(1+\epsilon, 1+\epsilon)$-approximation algorithm for the UHCMFL problem for the special case that the costs form a tree metric [31], $\forall \epsilon > 0$. Their algorithm is based on a dynamic programming approach. Clearly, the costs in the instance of the facility location problem in Section III do not form a tree metric. Hence, we can not use this result as the previous ones. Interestingly, using Fakcharoenphol *et al.*'s machinery [31], we can translate the above solution to obtain a $(\log |\mathcal{V}|, 1+\epsilon)$-approximation algorithm for general metrics. This translation requires polynomial time for the uniform capacities case and quasi-polynomial time for the general case. Finally, the work in [22] provided the first approximation algorithms that violate the capacities by a factor less than two and achieve a constant approximation ratio. The result is based on a reduction to a restricted version of the initial problem in a way that any $(O(1), 1+\epsilon)$-approximation algorithm for the restricted problem implies an $(O(1), 1+\epsilon)$-approximation algorithm for the initial problem, for $\epsilon \in \{1/2, 1/3\}$.

B. Approximation Ratios for the JRC-UR Problem

Although JRC-UR and UHCMFL problems are equivalent in terms of their optimal solution, the extension of approximation algorithms from one to the other is not straightforward. Theorem 2 describes the way that the bi-criteria bound changes when translating the solution to handle the JRC-UR case. Let us define:

$$c' = \frac{D(0.5+c)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - \sum_{n\in\mathcal{N}}(B_n/s)} \tag{12}$$

Then, we have the following theorem:

Theorem 2: For any (α, β)-bi-criteria approximation algorithm for the UHCMFL problem there is an $(\alpha + (\alpha-1)c', (\beta-1)|\mathcal{I}|+1)$-bi-criteria approximation algorithm for the JRC-UR problem, requiring the same computational complexity.

Proof: The overall traffic routed to an SBS $n \in \mathcal{N}$ by the real users is $\beta|\mathcal{I}|(B_n/s) - (|\mathcal{I}| - \lfloor S_n/s \rfloor)(B_n/s) - (\lfloor S_n/s \rfloor - 1)\lfloor B_n/s \rfloor$ in the worst case. This is because, each SBS corresponds to $|\mathcal{I}|$ facilities, each one of which has capacity equal to the capacity of the SBS. The virtual clients of the type $b'_{nj} \in \mathcal{B}_2$ and $b''_{nj} \in \mathcal{B}_3$, $\forall j$, must be served by the facilities a_{ni} $\forall i$, in any case, as explained in Lemma 3. That is the reason that we subtracted the traffic sent to them in the above expression. However, in reality, only B_n/s amount of data can be transmitted by each SBS. The fraction of the two values, after some computations, can be written as $(\beta-1)|\mathcal{I}|+1$.

Besides, let opt and $approx$ be the optimal and an approximation solution of the JRC-UR problem respectively. By Lemma 2, it holds that:

$$approx + D(0.5+c) \leq \alpha\,(opt + D(0.5+c)) \tag{13}$$

or equivalently:

$$approx \leq \left(\alpha + \frac{D(0.5+c)(\alpha-1)}{opt}\right) opt \tag{14}$$

Finally, it is:

$$opt \geq \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - \sum_{n\in\mathcal{N}}(B_n/s) \tag{15}$$

which completes the proof. ■

Fig. 4. Our algorithms operate in three stages. In Stage I, the MNO transforms the JRC-UR into the UHCMFL instance. In Stage II, it solves that instance by employing one of the algorithms in Table II. In Stage III, it maps the obtained solution to a solution for the JRC-UR instance based on the rules in Section III-A.

Theorem 2 combined with Algorithms 4 and 7 in Table II, provides two bi-criteria approximation algorithms for the JRC-UR problem. Corollary 1 describes this result:

Corollary 1: There exist a *polynomial time* (α_1, β_1)-bi-criteria approximation algorithm and a *quasi-polynomial time* (α_2, β_2)-bi-criteria approximation algorithm for the JRC-UR problem, for:

$$(\alpha_1, \beta_1) = (11 + 10c', |\mathcal{I}| + 1) \tag{16}$$

$$(\alpha_2, \beta_2) = (\log v + (\log v - 1)c', \epsilon|\mathcal{I}| + 1), \epsilon > 0 \tag{17}$$

where v is the size of the instance of the UHCMFL problem corresponding to the JRC-UR problem.

We can use the above bi-criteria solutions to perform caching and routing in our problem as it is described in Fig. 4. Note also that, as the bandwidth capacities of the SBSs may be violated by the above factors, the operator may need to endow the base stations with additional bandwidth capacity, to ensure the described approximation ratio. Nevertheless, in many cases, the operator is unwilling (or, incapable) to perform additional investments. Thus, the additional requests that reach an SBS will be rerouted to the MBS, further increasing its load. How much worse is the obtained result? Theorem 3 characterizes the worst case scenario in terms of the quality of the resulted solution.

Theorem 3: For any (α, β)-bi-criteria approximation algorithm for the UHCMFL problem there is an $(\alpha + (\alpha - 1)c')c''(\beta)$-approximation algorithm for the JRC-UR problem, requiring the same computational complexity, where:

$$c''(\beta) = \frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - \sum_{n\in\mathcal{N}}(B_n/s)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - ((\beta-1)|\mathcal{I}|+1)\sum_{n\in\mathcal{N}}(B_n/s)} \tag{18}$$

Proof: Let H_β be the number of requests routed to the SBSs and R_β be the number of requests routed to the MBS, according to the described reduction, when the capacities of the facilities are violated by a factor of β. In reality, all the requests beyond the capacities of the SBSs will be rerouted to the MBS, as the SBSs can not serve them. Let H be the number of requests served by the SBSs and R the number of requests served by the MBS after this rerouting. Then, it holds:

$$\begin{aligned}\frac{R}{R_\beta} &= \frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - H}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - H_\beta} \\ &= \frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - H}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - H\cdot((\beta-1)|\mathcal{I}|+1)} \\ &= \frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - \sum_{n\in\mathcal{N}}\lfloor\frac{B_n}{s}\rfloor}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) - \sum_{n\in\mathcal{N}}\lfloor\frac{B_n}{s}\rfloor\cdot((\beta-1)|\mathcal{I}|+1)}\end{aligned} \tag{19}$$

where the first equation holds because of the definition of the terms H, H_β, R and R_β which yields that: $H_\beta + R_\beta = H + R = \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki})$. The second equation is because of theorem 2: $H_\beta = H((\beta-1)|\mathcal{I}|+1)$. Finally, the last equation holds because after the rerouting of requests each SBS will only serve as many requests as its capacity allows. ■

Theorem 3 combined with the results in Table II, provides two approximation algorithms for the JRC-UR problem:

Corollary 2: There exists a *polynomial time* α_3-approximation algorithm and a *quasi-polynomial time* α_4-approximation algorithm, for the JRC-UR problem, for

$$\alpha_3 = (11 + 10c')c''(2) \tag{20}$$

$$\alpha_4 = (\log v + (\log v - 1)c')\, c''(1+\epsilon), \epsilon > 0, \tag{21}$$

where v is the size of the instance of the UHCMFL problem corresponding to the JRC-UR problem.

C. The Case of Uniform-Capacity SBSs

In this subsection, we focus on the special case of the JRC-UR problem where the SBSs have equal transmission capacities, i.e., $B_n = B,\ \forall n \in \mathcal{N}$. For example, assume that SBSs are of the same type, e.g., certain type of femtocells or picocells. However, the cache sizes can be different. We can map this problem to a certain UHCMFL problem in which all the capacities of the facilities are *equal* and exploit the improved approximation ratios that are known for this uniform capacity setting. According to the reduction described in Section III-A, for this special case of the problem, all the capacities of the facilities are equal, i.e., $C_j = B/s,\ \forall j \in \mathcal{A}$, except for the facility a_M, which has infinite capacity. Parameter a_M can be replaced by $\lceil\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}\lambda_{ki}/\lfloor B/s\rfloor\rceil$ facilities each one of capacity B/s. Clearly, the aggregate capacity of them suffices to serve all the demand of the clients of the form $b_{kij} \in \mathcal{B}_1,\ \forall k, i, j$ and the new instance is equivalent to the initial.

Based on the above, Table II provides the following approximation algorithms for the uniform-capacity JRC-UR problem. Corollary 3 describes the results:

Corollary 3: For the uniform-capacities JRC-UR problem, there exists an r_1-bi-criteria approximation algorithm, and an r_2-approximation algorithm, such that:

$$r_1 = (\alpha + (\alpha - 1)c', (\beta - 1)|\mathcal{I}| + 1) \tag{22}$$

$$r_2 = (\alpha + (\alpha - 1)c')\, c''(\beta) \tag{23}$$

$$\text{for: } (\alpha, \beta) \in \{(9, 4), (7.62, 4.29), (O(1), 2), (5, 2), (\log v, 1 + \epsilon), (10.173, 1.5), (30.432, 4/3)\}, \epsilon > 0,$$

where v is the size of the instance of the UHCMFL problem corresponding to the JRC-UR problem.

D. A Closer Look to the Approximation Ratio Values

In general, the values of the approximation ratios highly vary with the size of the UHCMFL instance $(|\mathcal{V}|)$, the number of SBSs (N) and their capacities (S_n, B_n), the number of files $(|\mathcal{I}|)$ and the number of user requests $(\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}\lambda_{ki})$. For example, as ϵ goes to zero, the approximation ratio of Algorithm 7 becomes $\log|\mathcal{V}| + (\log|\mathcal{V}| - 1)c'$. Replacing in (12),

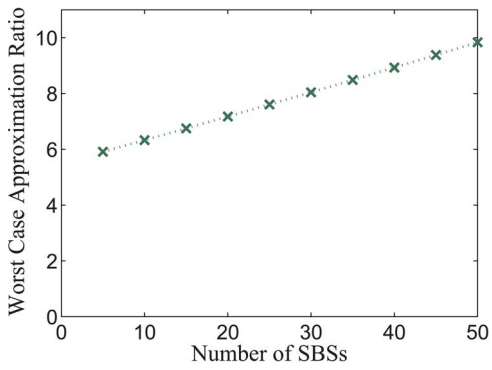

Fig. 5. The impact of the number of SBSs to the approximation ratio. System parameters: $|\mathcal{I}| = 100$ files, unit file size, $B_n/I = 0.05$, $S_n/I = 0.05$, $\forall n$, and 10,000 total user requests.

the value $D = \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}\lambda_{ki} + NB|\mathcal{I}| - NB$ that holds by definition, $B_n = B$, $\forall n \in \mathcal{N}$ and $c \to 0$, we can express c' as:

$$\begin{aligned} c' &= \frac{1}{2}\cdot\frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}\lambda_{ki} + NB|\mathcal{I}| - NB}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}\lambda_{ki} - NB} \\ &= \frac{1}{2}\left(1 + \frac{|\mathcal{I}|}{f-1}\right) \\ &= \approx \frac{1}{2}\left(1 + \frac{|\mathcal{I}|}{f}\right) \end{aligned} \tag{24}$$

where

$$f = \frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}\lambda_{ki}}{NB} \tag{25}$$

That is, the approximation ratio value scales proportionally to the fraction of the number of files $|\mathcal{I}|$ to the *congestion factor* f defined above. For example, Fig. 5 shows the approximation ratio for the system parameters: $|\mathcal{I}| = 100$, $s = 1$, $B_n/I = 0.05$, $S_n/I = 0.05$, $\forall n$, and $\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}(\lambda_{ki}) = 10{,}000$ and for $N \in \{5, 10, \ldots, 50\}$. Such set-ups are of importance as it is nowadays well-known that a small portion of content items is responsible for a large portion of the traffic (e.g., viral video items) [28], [32]. Finally, interestingly enough, we notice that the ratio increases only 40% when the number of SBSs increases 500%.

Although not particularly tight in general, the derived approximation ratios constitute the *first* result in the area of approximating a joint caching and routing problem for small cell networks. Additionally, the proposed mapping ensures that any improved approximation ratios for the facility location problem that may appear in the near future will be applicable in this caching problem (using the updated Table II). More importantly, as we show with an extensive numerical study in the next section, the proposed algorithms operate much better than the worst case conditions indicate in realistic network architectures.

V. Performance Evaluation

In this section, we present the numerical results of the experiments that we have conducted to evaluate our derived theoretical results. Specifically, using realistic system settings we characterize the performance improvements offered by one of the proposed algorithms over conventional caching schemes, as well as its performance gap to the optimal solution. Please notice that we have made a complete parameter analysis as we have explored how the variation in all system parameters affects the performance of the algorithm compared to other similar schemes.

A. Simulation Setup and Methodology

We compare the performance of the four following schemes:

(1) *Greedy*: The naive approach according to which each SBS caches the most popular files based on the requests of the nearby users independently from the others. When a request is generated, it is routed to the nearest SBS that has stored a copy of the associated file.

(2) *Iterative* [7]: It starts with all the caches empty. At each iteration, it places the file to a non-full cache that yields the lowest value of the objective function in (3). The algorithm terminates when all the caches become full. When a request is generated, it is routed to the nearest SBS that has stored a copy of the associated file.

(3) *Facility*: The routing and caching policies are jointly derived by solving the instance of the facility location problem using Algorithm 1 in Table II, as described in detail in Section IV, and it is shown in Fig. 4.

(4) *Optimal*: The optimal solution of the JRC-UR problem found through exhaustive search. Since its running time is unacceptable large, i.e., in the scale of days, using realistic system settings, Optimal is only used as a benchmark for gauging the performance of the proposed solutions and determine if there is still room for improvement.

The performance criterion we use is the total number of requests that reach the MBS $(MBS\ load)$. To describe in detail the performance improvements of the Facility scheme compared to its alternatives, we also depict the normalized difference between the MBS load achieved by any of the first three schemes and the Optimal $(MBS\ load\ difference)$. Formally, the MBS load difference of the Greedy algorithm is defined as:

$$\frac{MBS\_load_{Greedy} - MBS\_load_{Optimal}}{MBS\_load_{Optimal}} \tag{26}$$

where MBS_load_{scheme} denotes the MBS load achieved by the associated scheme, i.e., the value $\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}} y^k_{Mi}$. A similar definition holds for the Iterative and the Facility Scheme.

Throughout, we consider a single MBS serving a circular-shaped cell with radius 350 meters (typical of urban macrocell [7]). We assume that $N = 16$ SBSs are randomly deployed within it, each one having a communication range of 80 meters. The system supports the service of a rich collection of $I = 1{,}000$ unit-sized files. Unless otherwise specified, a large number of $K = 1{,}000$ mobile users are uniformly placed in random statistically independent positions in the cell. Each user requests one file based on the zipf law with shape parameter $z = 0.8$ [33], i.e., the probability that a request is for the jth most popular file is: $j^{-z}/\sum_{i=1}^{|\mathcal{I}|} i^{-z}$. Each SBS n is endowed with a cache of size $S_n = S$, $\forall n \in \mathcal{N}$ that is equal to 3% of the entire file set size. Finally, its bandwidth $B_n = B$, $\forall n \in \mathcal{N}$ suffices for transmitting 5% of the entire file set.

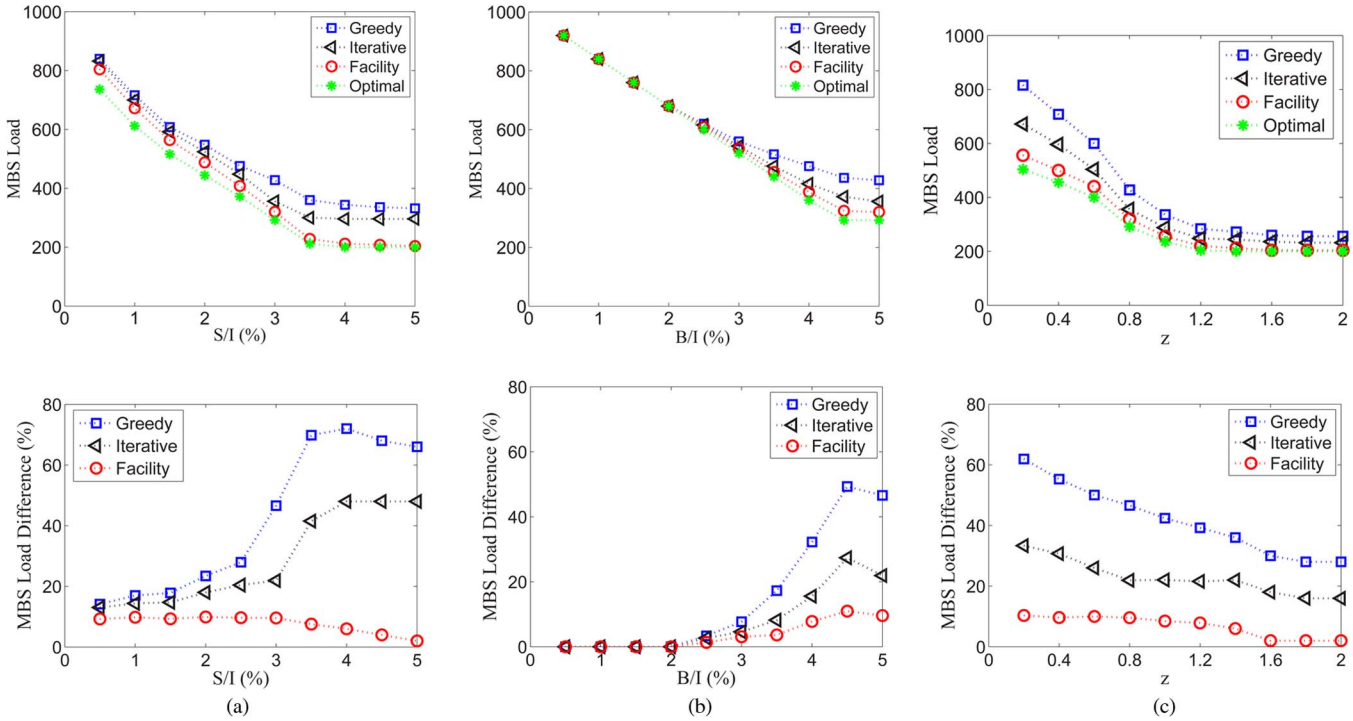

Fig. 6. Performance comparison between Greedy, Iterative, Facility and Optimal scheme for various values of (a) the cache size, (b) the bandwidth capacity per SBS and (c) the zipf parameter of the popularity distribution of the files.

B. Parameter Impact Analysis

1) Impact of the Cache Sizes: Fig. 6(a) compare the performance of the discussed schemes as a function of the cache size S of each SBS. Parameter S varies in our simulation from 0.5% to 5% of the entire file set size. As expected, increasing the available cache space, decreases the MBS load for all the schemes, as more files are cached at the SBSs. More importantly, as S increases the performance of Facility scheme comes very close to the optimal one. Even for low values of S, the Facility scheme operates very close to the Optimal (less than 10% worse), and far better than the worst case conditions indicate. Besides, the Facility scheme provides significant performance gains, up to 38%, over the Greedy and Iterative schemes. To elaborate on this, we observe that although the Iterative scheme performs the cache placement more efficiently than Greedy (since it places the files in multiple stages rather than simultaneously), both schemes fail to appropriately route the user requests to the SBSs. This is because, they both ignore the bandwidth limitations of the SBSs (bandwidth-agnostic).

2) Impact of the Transmission Bandwidth Capacities: We analyze the impact of the transmission bandwidth capacities on the algorithms' performance in Fig. 6(b). We vary the bandwidth capacity per SBS B from 0.5% to 5% of the entire file set size. As expected, increasing B, decreases the MBS load, since the SBSs can serve more requests. We observe that for low values of B, i.e., when the system is in overloaded conditions, the performance of the three schemes is similar. This is because, in these cases, simply caching the S most popular file at each SBS suffices to fully utilize its bandwidth capacity for almost all the SBSs. Interestingly, the performance gap between Facility and Optimal scheme increases as B increases in the range of 0 to 10%. This is because, as explained in Section IV, the solution of the facility location problem may violate the bandwidth capacities of the SBSs, and redirecting the extra requests to the MBS further increases its load. This is more crucial for high values of B. Finally, we note that Facility scheme consistently outperforms the Greedy and Iterative schemes, a gap that increases with B in the range of 0% to 25%.

3) Impact of the File Request Pattern: We explore the impact of the steepness of the file request pattern on the algorithms' performance in Fig. 6(c). That is, how do the algorithms perform as a small number of files become very popular (e.g., in the context of a social network [27]). Namely, we vary the shape parameter z of the file popularity from the value 0.2 to 2. We observe that as z increases, the MBS load decreases for all the schemes, reflecting the well known fact that caching effectiveness improves as the popularity distribution gets steeper. Besides, as z increases the performance gap between each pair of the discussed algorithms is shrinking. This is because, when z is high, the vast majority of user requests refer to a small number of files. Clearly, caching the above files provides significant benefits to the provider. To conclude, our algorithm achieves a performance that is up to 31% better than the Greedy, up to 17% better than Iterative, and less than 10% worse than the optimal one.

4) Impact of the Demand Heterogeneity: To capture the fact that different demand volumes may appear across different regions in the cell (e.g., in the city center, a university campus or a less popular venue), we synthesize a variety of scenarios that differ in the way that user demand is generated and investigate the impact of our algorithms in each. Specifically, we allow each user to generate an integer number of requests (not necessarily one) that is uniformly picked from

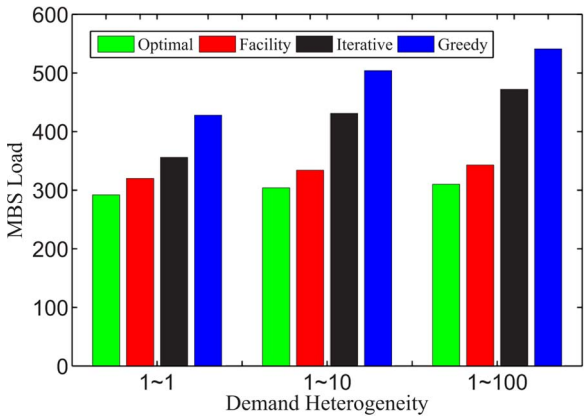

Fig. 7. The impact of the heterogeneity of demand on the performance of the Optimal, Facility, Iterative, and Greedy scheme.

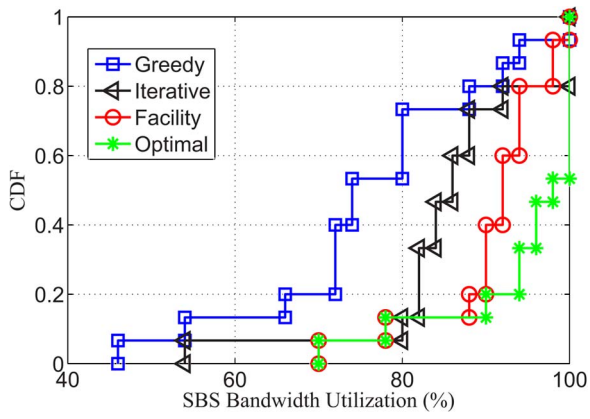

Fig. 8. The cumulative distribution function (CDF) of the per SBS bandwidth utilization achieved by the Greedy, Iterative, Facility and Optimal scheme.

a set $[l, u]$, and denote this with $l \sim u$. Then, we start placing the users in uniform and independent positions in the cell and (to ensure a fair comparison with Fig. 6) we stop as soon as the total number of requests reaches the value 1000. Fig. 7 compares the performance of the four presented schemes for the scenarios that $l \sim u$ is $1 \sim 1$, $1 \sim 10$ and $1 \sim 100$. The MBS load increases more-or-less for all the schemes as the request heterogeneity increases. This can be explained by the fact that, for heterogeneous demand, certain SBSs that are close to users with large demand volumes exhibit very high demand (that inevitably remains partially unserved), while the rest are underloaded. Interestingly enough, we notice that the performance of the bandwidth-agnostic schemes (i.e., Iterative and Greedy) is rather sensitive to the heterogeneity of the demand, while that of the Facility is more *stable*. Specifically, we see that the MBS load when applying the Facility scheme increases only 7.2%, while for the two heuristic schemes, the increase is up to 32.58%.

Finally, we analyze the bandwidth capacity utilization for each SBS in Fig. 8, for $l \sim u = 1 \sim 1$, $S/I = 3\%$, $B/I = 5\%$ and $z = 0.8$. Interestingly, we uncover examples of SBSs with extremely low utilization under the operation of the Greedy and Iterative schemes. We suspect these SBSs to be located at remote areas of the cell, thus considered less preferable for service by most of the users than the central ones. On the contrary, Facility scheme mimics the way that content is cached and delivered to the users according to the Optimal scheme, and hence manages to utilize most of the SBS bandwidth to serve the anticipated demand.

VI. Related Work

Optimized caching policies have long been investigated for wireline networks. Korupolu *et al.* [34] developed a polynomial-time optimal algorithm for the hierarchical caching problem. Their solution is based on a reduction to minimum cost flow problem. Subsequently, Borst *et al.* [35] developed approximation caching algorithms aiming to minimize the bandwidth cost. In our previous work [36], we presented a polynomial-time optimal algorithm for certain instances of that problem, based on a reduction to a matching problem. However, all the above results are based on the assumption that the link capacities are never the bottleneck for the content delivery to the users. In contrast, we consider here the realistic case that hard bandwidth constraints limit the transmission of the cached items at the base stations.

A method to further increase the benefits of caching is to jointly design caching and routing policies. Related works [16] and [37] studied the joint caching and request routing problem in CDNs, while [15] provided a similar study for IPTV networks. These works employ the Lagrangian relaxation method and use iterative algorithms to reach a solution that satisfies a certain optimality criterion. However, there is no guarantee about the efficiency of the obtained result nor the running time of the algorithms, which are only evaluated numerically. Besides, these models assume that request routing is fractional, i.e., an arbitrary portion of each user requests can be served by different base stations. In contrast, in our model, the routing variables are discrete and the proposed approximation algorithms require fixed running time.

However, caching in small cell networks is a novel approach [7]–[10], [38]–[45] and differs significantly from the above architectures. Namely, there are multiple routes to the users, i.e., the last hop link can be from the SBSs to the users or from the MBS to the users. Additionally, the SBSs, unlike the routers and servers in respective wireline networks, are (typically) connected via *low-speed* backhaul links to the core network and hence can not serve each others requests. Hence, the joint routing and caching policy design problem must be revisited. Previous works have designed optimal caching policies by assuming that the transmission bandwidth of the base stations suffices (non-congestible) to serve all the content requests. The proposed schemes employ discrete/convex optimization techniques [7], they are based on the content centric networking [8] or the "stable-matching" [38] concept, or they are designed with mobility-awareness [39] (users associate with multiple SBSs as they move) and multicast-awareness [40] (requests for the same file generated at nearby times are aggregated and served via a single multicast transmission). Besides, they leverage a wide range of technologies, such as: advanced video encoding [41] (for different video frame-rates/SNR qualities/resolutions), learning [10], [43] (for partially known content demand),

cooperative MIMO [44] (for interference mitigation), and auction mechanisms [45] (for utility maximization of the small-cell owners). However, when these links are congestible, either due to limited resources or because user demand is massive, the caching decisions should be designed with respect to routing policies.

Our recent work in [46] extended the above model for the scenario that allows for selectively fetching requested files on demand through the (capacitated) backhaul links of the SBSs. While it tackles a similar joint routing and caching problem, the objectives are different (i.e., it explores the energy-delay tradeoff of alternative video encoding schemes), and the result of the optimization is evaluated numerically. In contrast, the aim of this work is to present a novel optimization framework for the problem of maximizing the fraction of user requests absorbed by the SBSs which are connected by low-speed backhaul links. Besides, this is the first work, building on our initial study [1], that identifies this extensive relation of facility location problems and caching in small cell networks.

VII. Conclusion

In this paper, we considered cellular networks enhanced with small cell base stations (SBSs) and we studied the problem of minimizing the user content requests that need to be routed to the macrocell base stations. This is a problem of increasing importance as currently the explosion of mobile data traffic challenges mobile network operators. Unlike previous works, we explicitly took into account both the constrained storage and transmission bandwidth capacities of the SBSs. This scenario is particularly important as SBSs are often deployed in highly dense and congested urban areas.

We formulated the joint routing and caching policy design problem (JRC-UR), which is an NP-hard problem, and we introduced a novel approximation framework that allowed its analytical study. Our methodology builds upon a connection which we identified among the JRC-UR problem and the facility location variant problem known as UHCMFL. This enabled the derivation of a set of polynomial time algorithms with approximation guarantees. Numerical results based on realistic system settings, illustrated the performance benefits of our approach which more often than not, performs very close to the optimal solution and much better than the worst case scenario, i.e., the approximation bound. We believe that this work reveals a particularly interesting connection between caching problems in small cell networks and facility location problems, and paves the road for exploiting the broad literature that is available for the latter.

Acknowledgment

This work was supported by the FP7 NoE project EINS (FP7–288021) co-financed by Greece and the European Union (European Social Fund) through the Operational Program "Education and Lifelong Learning" – NCRF 2007–2013. The first author would like to acknowledge the "Alexander S. Onassis", Public Benefit Foundation, Greece for providing a scholarship.

References

[1] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation caching and routing algorithms for massive mobile data delivery," in *Proc. IEEE Globecom*, 2013, pp. 3534–3539.
[2] "Cisco visual networking index: Global mobile data traffic forecast update, 2012–2017," San Jose, CA, USA, Feb. 2013.
[3] "Ericsson mobility report," Stockholm, Sweden, Jun. 2014. [Online]. Available: http://www.ericsson.com/mobility-report
[4] J. G. Andrews, "Seven ways that HetNets are a cellular paradigm shift," *IEEE Commun. Mag.*, vol. 51, no. 3, pp. 136–144, Mar. 2013.
[5] Intel, "Rethinking the small cell business model," White Paper, 2011.
[6] "Backhaul technologies for small cells: Use cases, requirements and solutions," Dursley, U.K., Tech. Rep./White Paper, 2013.
[7] 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, pp. 1107–1115.
[8] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5G systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 131–139, Feb. 2014.
[9] A. F. Molisch, G. Caire, D. Ott, J. R. Foerster, D. Bethanabhotla, and M. Ji, "Caching eliminates the wireless bottleneck in video-aware wireless networks," *arXiv preprint arXiv:1405.5864*, 2014.
[10] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactice caching in 5G wireless networks," *IEEE Comm. Mag.*, vol. 52, no. 8, pp. 82–89, Aug. 2014, SI on Context Awareness.
[11] Ubiquisys Press Release, Saguna Networks and Ubiquisys Partner to Provide Content and Application Acceleration for Small Cells, Feb. 2013. [Online]. Available: http://www.ubiquisys.com/small-cells-media-press-releases-id-313.htm
[12] AltoBridge Press Release, Data-at-the-Edge (DatE) Platform, 2012. [Online]. Available: http://www.altobridge.com/data-at-the-edge
[13] ThinkSmallCell Portal, Small Cell Backhaul Data Caching Reduces Backhaul Costs for Small Cells and WiFi, May 2013. [Online]. Available: http://www.thinksmallcell.com/Backhaul/data-caching-reduces-backhaul-costs-for-small-cells-and-wi-fi.html
[14] A. Ghosh, J. Zhang, J. Andrews, and R. Muhamed, *Fundamentals of LTE*, *Prentice Hall Communications Engineering and Emerging Technologies Series*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2010.
[15] J. Dai, Z. Hu, B. Li, J. Liu, and B. Li, "Collaborative hierarchical caching with dynamic request routing for massive content distribution," in *Proc. IEEE INFOCOM*, 2012, pp. 2444–2452.
[16] T. Bektas, J.-F. Cordeau, E. Erkut, and G. Laporte, "Exact algorithms for the joint object placement and request routing problem in content distribution networks," *Comput. Oper. Res.*, vol. 35, no. 12, pp. 3860–3884, Dec. 2008.
[17] D. B. Shmoys, E. Tardos, and K. Aardal, "Approximation algorithms for facility location problems," in *Proc. ACM STOC*, 1997, pp. 265–274.
[18] M. Korupolu, C. Plaxton, and R. Rajaraman, "Analysis of a local search heuristic for facility location problems," in *Proc. SODA*, 1998, pp. 1–10.
[19] J. Zhang, B. Chen, and Y. Ye, "A multi-exchange local search algorithm for the capacitated facility location problem," *Math. Oper. Res.*, vol. 30, no. 2, pp. 389–403, May 2005.
[20] A. Aggarwal *et al.*, "A 3-approximation for facility location with uniform capacities," in *Proc. IPCO*, 2010, pp. 149–162.
[21] M. Bateni and M. Hajiaghayi, "Assignment problem in content distribution networks: Unsplittable hard-capacitated facility location," *ACM Trans. Algorithms*, vol. 8, no. 3, p. 20, 2012.
[22] B. Behsaz, M. R. Salavatipour, and Z. Svitkina, "New approximation algorithms for the unsplittable capacitated facility location problem," in *Proc. SWAT*, 2012, pp. 237–248.
[23] L. Duan, J. Huang, and B. Shou, "Investment and pricing with spectrum uncertainty: A cognitive operator's perspective," *IEEE Trans. Mobile Comput.*, vol. 10, no. 11, pp. 1590–1604, Nov. 2011.
[24] Y. Kishiyama, A. Benjebbour, H. Ishii, and T. Nakamura, "Evolution concept and candidate technologies for future steps of LTE-A," in *Proc. IEEE ICCS*, 2012, pp. 473–477.
[25] W. Cheung, T. Quek, and M. Kountouris, "Throughput optimization, spectrum allocation, access control in two-tier femtocell networks," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 3, pp. 561–574, Apr. 2012.
[26] D. Astely *et al.*, "LTE: The evolution of mobile broadband," *IEEE Commun. Mag.*, vol. 47, no. 4, pp. 44–51, Apr. 2009.
[27] X. Bao, Y. Lin, U. Lee, I. Rimac, and R. R. Choudhury, "DataSpotting: Exploiting naturally clustered mobile devices to offload cellular traffic," in *Proc. IEEE INFOCOM*, 2013, pp. 420–424.

[28] M. Zinka, K. Suhb, Y. Gua, and J. Kurose, "Characteristics of youtube network traffic at a campus network—Measurements, models, implications," *Comput. Netw.*, vol. 53, no. 4, pp. 501–514, Mar. 2009.
[29] K. Son, H. Kim, Y. Yi, and B. Krishnamachari, "Base station operation and user association mechanisms for energy-delay tradeoffs in green cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 8, pp. 1525–1536, Sep. 2011.
[30] D. P. Williamson and D. B. Shmoys, *The Design of Approximation Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2011.
[31] Y. Bartal, "Probabilistic approximations of metric spaces and its algorithmic applications," in *Proc. IEEE Symp. Found. Comput. Sci.*, 1996, pp. 184–193.
[32] S. Woo *et al.*, "Comparison of caching strategies in modern cellular backhaul networks," in *Proc. ACM Mobisys*, 2013, pp. 319–332.
[33] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Transactions on Networking*, vol. 16, no. 6, pp. 1447–1460, Dec. 2008.
[34] M. Korupolu, C. Plaxton, and R. Rajaraman, "Placement algorithms for hierarchical cooperative caching," in *Proc. ACM-SIAM SODA*, 1999, pp. 586–595.
[35] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution network," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[36] K. Poularakis and L. Tassiulas, "Optimal cooperative content placement algorithms in hierarchical cache topologies," in *Proc. CISS*, 2012, pp. 1–6.
[37] J. W. Jiang, S. Ioannidis, L. Massoulie, and F. Picconi, "Orchestrating massively distributed CDNs," in *Proc. ACM CoNEXT*, 2012, pp. 133–144.
[38] K. Hamidouche, W. Saad, and M. Debbah, "Many-to-many matching games for proactive social-caching in wireless small cell networks," in *Proc. Wiopt, WNC3 Workshop*, 2014, pp. 569–574.
[39] K. Poularakis and L. Tassiulas, "Exploiting user mobility for wireless content delivery," in *Proc. IEEE ISIT*, 2013, pp. 1017–1021.
[40] K. Poularakis, G. Iosifidis, V. Sourlas, and L. Tassiulas, "Multicast-aware caching for small-cell networks," in *Proc. IEEE WCNC*, 2014, pp. 2330–2335.
[41] P. Ostovari, A. Khreishah, and J. Wu, "Cache content placement using triangular network coding," in *Proc. IEEE WCNC*, 2013, pp. 1375–1380.
[42] E. Bastug, J. L. Guenego, and M. Debbah, "Proactive small cell networks," in *Proc. 20th ICT Conf.*, 2013, pp. 1–5.
[43] P. Blasco and D. Gunduz, "Learning-based optimization of cache content in a small cell base station," *arXiv preprint arXiv:1402.3247*, 2014.
[44] A. Liu and V. K. N. Lau, "Mixed-timescale precoding and cache control in cached MIMO interference network," *IEEE Trans. Signal Process.*, vol. 61, no. 24, pp. 6320–6332, Dec. 2013.
[45] J. Yue, B. Yang, C. Chen, X. Guan, and W. Zhang, "Femtocaching in video content delivery: Assignment of video clips to serve dynamic mobile users," *Comput. Commun.*, vol. 51, pp. 60–69, Sep. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2014.05.010
[46] K. Poularakis, G. Iosifidis, A. Argyriou, and L. Tassiulas, "Video delivery over heterogeneous cellular networks: Optimizing cost and performance," in *Proc. IEEE INFOCOM*, 2014, pp. 1078–1086.

Konstantinos Poularakis received the Diploma and the M.S. degree in computer and communications engineering from University of Thessaly, Greece, in 2011 and 2013, respectively. Currently, he is pursuing the Ph.D. candidate in the Department of Electrical & Computer Engineering, University of Thessaly, Greece. His research interests lie in the broad area of network optimization with emphasis on mobile data caching and network coding for distributed storage. He has been honored with several awards and scholarship during his studies, from sources including the Greek State Scholarships foundation (IKY) and the Center for Research and Technology Hellas (CERTH). He is also a scholar of "Alexander S. Onassis Public Benefit Foundation".

George Iosifidis received the Diploma in electronics and telecommunications engineering from the Greek Air Force Academy, in 2000, and the M.S. and Ph.D. degrees in electrical engineering from the University of Thessaly, Greece, in 2007 and 2012, respectively. Currently, he is a Post-doc Researcher at the University of Thessaly and CERTH, Greece. His research interests lie in the broad area of network optimization and network economics.

Leandros Tassiulas (S'89–M'91–SM'06–F'07) received the Diploma in electrical engineering from the Aristotelian University of Thessaloniki, Greece, in 1987, and the M.S. and Ph.D. degrees from the University of Maryland, in 1989 and 1991, respectively. Since 2002, he is a Professor in the Department of Electrical and Computer Engineering, University of Thessaly. He has held positions as Assistant Professor at Polytechnic University New York (1991–1995), Assistant and Associate Professor at University of Maryland (1995–2001), and Professor at University of Ioannina, Greece (1999–2001). His research interests are in the field of computer and communication networks with emphasis on fundamental mathematical models, architectures and protocols of wireless systems, sensor networks, high-speed Internet, and satellite communications. Dr. Tassiulas received a National Science Foundation (NSF) Research Initiation Award in 1992, an NSF CAREER Award in 1995, an Office of Naval Research Young Investigator Award in 1997, and a Bodosaki Foundation award in 1999. He also received the INFOCOM 1994 best paper award and the INFOCOM 2007 achievement award.