Πανεπιστήμιο Θεσσαλίας

Σχολή: Επιστήμη και Τεχνολογία Η/Υ Τηλ/νιών και Δικτύων

Μεταπτυχιακή Διπλωματική Εργασία

21/03/2013

“Κωδικοποίηση και Αποθήκευση σε Δίκτυα Διανομής Περιεχομένου”

Κωνσταντίνος Πουλαράκης

Επιβλέποντες:

Λέανδρος Τασιούλας

Ιορδάνης Κουτσόπουλος

Ιωάννης Κατσαβουνίδης

Αφιερωμένο στην οικογένεια μου,

Νίκο, Χριστίνα και Στέργιο

και στον Μιχαήλ,

Περίληψη

Σήμερα είμαστε μάρτυρες μιας άνευ προηγουμένου σε όλο τον κόσμο ανάπτυξης των κινητών δεδομένων κίνησης που αναμένεται να συνεχιστεί με ετήσιο ποσοστό του 78 τοις εκατό κατά τα επόμενα χρόνια, φτάνοντας τα 10,8 exabytes / μήνα μέχρι το 2016. Οι εξελίξεις αυτές οδηγούν την ερευνητική κοινότητα να διερευνήσει τρόπους για την αύξηση της φασματικής απόδοσης των κυψελοειδών δικτύων. Ο πλέον υποσχόμενος τρόπος για να επιτευχθεί αυτό είναι η ανάπτυξη των μικρών σταθμών βάσης κοντά στους χρήστες που χειρίζονται ένα κλάσμα της κυκλοφορίας αντί για το συμβατικό σταθμό βάσης της κυψέλης. Το μειονέκτημα αυτής της προσέγγισης είναι η υψηλή τιμή των ζεύξεων που ενώνουν τους μικρότερους σταθμούς με τον κεντρικό σταθμό της κυψέλης. Πρόσφατα ερευνητικά αποτελέσματα έδειξαν ότι εξοπλίζοντας αυτούς τους σταθμούς βάσης με δυνατότητες αποθήκευσης μειώνει το παραπάνω κόστος.

Σε αυτή την εργασία, επιτρέπουμε τη χρήση κωδικοποίησης κατά την αποθήκευση των αρχείων στους σταθμούς βάσης. Αυτό σημαίνει ότι κωδικοποιημένες εκδόσεις του περιεχομένου αποθηκεύονται στις κρυφές μνήμες των σταθμών αντί των πρωτότυπων πακέτων δεδομένων. Με τη χρήση ενός κατάλληλου κώδικα, η επιτυχημένη παράδοση περιεχομένου επιτυγχάνεται όταν το συνολικό ποσό των ληφθέντων δεδομένων από τους σταθμούς βάσης που συναντά ο χρήστης εντός της προθεσμίας είναι τουλάχιστον ίσο με το μέγεθος του αιτούμενου αρχείου. Εάν το ποσό της πληροφορίας το οποίο λάβει ο χρήστης, δεν αρκεί για να ανακτήσει το αρχείο που ζητήθηκε, ο κεντρικός σταθμός βάσης της κυψέλης εξυπηρετεί το αίτημα. Η τελευταία ενέργεια είναι η πλέον δαπανηρή για τον διαχειριστή του συστήματος.

Στο πρώτο κεφάλαιο, θα επικεντρωθούμε στο παραπάνω πρόβλημα της κατανομής του αποθηκευτικού χώρου στους σταθμούς βάσης της κυψέλης. Στόχος μας είναι να ελαχιστοποιήσουμε το κλάσμα των αιτήσεων για περιεχόμενο που εξυπηρετούνται από τον κύριο σταθμό βάσης του συστήματος. Η συνολική ποσότητα του περιεχομένου που μπορεί να αποθηκευτεί σε ένα σταθμό βάσης περιορίζεται από τη χωρητικότητα της κρυφής μνήμης του. Δείχνουμε πώς μπορούμε να εκμεταλλευτούμε την κινητικότητα των χρηστών για την αποτελεσματικότερη λήψη αποφάσεων αποθήκευσης. Υποθέτουμε ότι το πρότυπο της κινητικότητας των χρηστών ακολουθεί ένα διακριτού χρόνου Markov στοχαστικό μοντέλο και προτείνουμε ένα νέο προσεγγιστικό αλγόριθμο.

Στο δεύτερο κεφάλαιο, μελετούμε την ρεαλιστική περίπτωση κατά την οποία απαιτείται μαζική παράδοση δεδομένων. Έτσι, οι περιορισμοί χωρητικότητας των ζεύξεων αυτών έρχονται στο προσκήνιο. Διατυπώνουμε το πρόβλημα της από κοινού αποθήκευσης και δρομολόγησης των αιτήσεων των χρηστών στους σταθμούς βάσης. Δείχνουμε ότι το πρόβλημα αυτό είναι ισοδύναμο με ένα από τα προβλήματα της ευρύτερης περιοχής των “facility location problems”. Η ισοδυναμία αυτή μας επιτρέπει να εξάγουμε μία οικογένεια προσεγγιστικών αλγορίθμων για το αρχικό πρόβλημα.

Το κύριο σώμα της διπλωματικής εργασίας ακολουθεί (ως παράρτημα) στην αγγλική γλώσσα

CODING AND STORAGE IN CONTENT DISTRIBUTION NETWORKS

Konstantinos Poularakis

University of Thessaly, Volos, Greece.

kopoular@inf.uth.gr

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 has been proposed in order to decrease the capacity -and hence the cost- of the backhaul links that connect these base stations with the core network. However, deriving the optimal caching policy remains a challenging open problem.

In chapter 1, we allow coding in storage decisions at the base stations. This means that encoded versions of the content are stored at the caches instead of the raw data packets. By using an appropriate code, successful content delivery occurs when the total amount of the downloaded data is at least the size of the original file. We then present a distributed approximation algorithm based on a Markov chain analysis. We evaluate the proposed scheme using traces of a data set of real mobile users.

In chapter 2, we consider the scenario that massive content delivery is required and formulate the joint routing and caching problem aiming to maximize the fraction of content requests served by the deployed base stations. We present a novel approximation framework based on a reduction to a well-known variant of the facility location problem. This allows us to exploit the rich literature in facility location problems, in order to establish bounded approximation algorithms for our problem.

Chapter 1:
Exploiting User Mobility for Wireless Content Delivery

Abstract

We consider the problem of storing segments of encoded versions of content files in a set of base stations located in a communication cell. These base stations work in conjunction with the main base station of the cell. Users move randomly across the space based on a discrete-time Markov chain model. At each time slot each user accesses a single base station based on it's current position and it can download only a part of the content stored in it, depending on the time slot duration. We assume that file requests must be satisfied within a given time deadline in order to be successful. If the amount of the downloaded (encoded) data by the accessed base stations when the time deadline expires does not suffice to recover the requested file, the main base station of the cell serves the request. Our aim is to find the storage allocation that minimizes the probability of using the main base station for file delivery. This problem is intractable in general. However, we show that the optimal solution of the problem can be efficiently attained in case that the time deadline is small. To tackle the general case, we propose a distributed approximation algorithm based on large deviation inequalities. Systematic experiments on a real world data set demonstrate the effectiveness of our proposed algorithms.

I. Introduction

Today we are witnessing an unprecedented worldwide growth of mobile data traffic that is expected to continue at an annual rate of 78 percent over the next years, reaching 10.8 exabytes/month by 2016 [10]. These developments lead the research community to investigate ways for increasing area spectral efficiency of cellular networks. The most

promising way to achieve this is the deployment of small base stations near to the users that handle a fraction of wireless content traffic in place of the conventional base station of the cell. The drawback of this approach is the high price of the required backhaul to the main base station. The recent work in [1] showed that equipping these base stations with storage capabilities alleviates the backhaul cost. However, little work has been done on exploiting the user mobility in taking storage management decisions in these systems.

A user of these systems is connected to the base station that is currently in communication range based on it's geographical position. The dense spatial deployment of the base stations highlights the scenario that mobile users connect to more than one base stations as they move over time. This transition can happen in a *short time period* of some minutes taking into account the typical values of cell radius $(\approx 400m)$ and the deployment of some decades of base stations in the cell. Thus, only a *fraction* of the requested data may be downloaded by the connected base station. Besides, depending on the user preferences, it may be acceptable for it to wait a time period until the requested file is delivered. Thus, the user can fetch the entire content file that wishes from different base stations that encounters as it moves within a given time deadline.

In this work, we allow coding in storage decisions at the base stations. This means that encoded versions of the content are stored at the caches instead of the raw data packets. By using an appropriate code, successful content delivery occurs when the total amount of the downloaded data by the encountered base stations within the time deadline is at least the size of the requested file [4]. If the amount of the downloaded data when the time deadline expires does not suffice to recover the requested file, the main base station of the cell serves the request.

In this chapter, we focus on the above storage allocation problem at the base stations of the cell. Our goal is to minimize the fraction of file requests that are served by the main base station. The total amount of content stored at a base station is upper bounded by the capacity of it's cache. Traditionally, storage allocation at a cache node is performed

based on the average content popularity near that node. In contrast, we *exploit the user mobility in storage decision taking*. We assume that the user mobility pattern follows a discrete-time Markov model. This is a realistic assumption, as the future position of a user highly depends on it's current position. Thus, the user movement within a given time deadline can be represented by a random walk on a Markov chain. The probabilities of transition can be efficiently derived using learning mechanisms [7].

Our work builds upon the *Femtocaching architecture* proposed in [1] by exploiting user mobility statistics for storage allocation decision taking. The technical contributions of this chapter can be summarized as follows:

- Specifying the coded storage allocation problem.
- Presenting an *optimal* solution of the problem for short time deadline based on branch and bound algorithms.
- Proposing a *distributed* approximation algorithm for arbitrarily large time deadline: We minimize an appropriate probability bound.
- Evaluating the proposed schemes: We use *traces* of a data set of real mobile users.

The remainder of the chapter is organized as follows: Section II presents the related work and Sec. III presents the system model and the problem formulation. In Sec. IV we show that the problem is tractable for short time deadline case. Sec. V develops a distributed approximation algorithm by minimizing an appropriate probability bound. In Sec. VI we present our evaluation results. A summary concludes the chapter in Sec. VII.

II. Related work

The problem of storing segments of encoded versions of content in a distributed storage system is a well studied one in literature. Ntranos et al. [2] studied the above problem aiming to maximize the probability of recovery of the content after a random set of nodes fail given a total storage budget. Their work generalizes the results in [4], where homogeneous reliability parameters were assumed. Furthermore, recent works in [3] and [5]

studied the above problem in a delay tolerant network setting assuming that data recovery must be achieved within a given time deadline.

However, the methodologies used in all these works depend on the assumption of independent access of the data stored at the nodes. Thus, they are suitable for representing node failures scenarios or node encounters in delay-tolerant networks. In contrast, our work assumes that data access follows a Markov chain random model, that naturally represents the user movement in a cellular network. Besides, most of the existing work in the area simply assumes that as long as a user contacts a storage node, the complete requested data can be downloaded [3]. In contrast, we consider the realistic case that contact duration limits bottleneck the data transmission. Finally, our work builds upon a novel network architecture [1] by exploiting user mobility statistics for strategic data placement.

III. System model and problem formulation

We consider a single cell in which the main base station (MBS) serves the content requests of the mobile users that lie in it. A set of n smaller base stations are geographically deployed in the cell. Following the notation in [1] we name these base stations as $helpers$. Each helper h is endowed with a cache of size $|C_h|$. Let O denote a static collection of $|O|$ content files of sizes $|O_i|$, $i = 1, ..., |O|$. We consider the case that the coverage areas of the helpers are non-overlapping. Thus, a user is in communication range with the nearest helper each time. We denote by $P_{i/h}$ the probability that a user generated request that happens in the area around helper h, corresponds to file O_i.

As the users move in space they encounter different helpers over time. In order to represent the system evolution we use a time-homogeneous discrete-time Markov chain M of n states named as $X_1, ..., X_n$, where state X_h denotes that a specific user accesses helper h. The initial probability distribution of the chain is denoted by P_{init}. Clearly, a large value of $P_{init}(h)$ means that the area around helper h is highly populated. The probability of transition $M_{h',h}$, denotes the probability that a specific user encounters the helper h

at a time slot, given that the previous time slot the user was connected to the helper h'. Typically, pairs of helpers that are neighbors will present higher probabilities of transition than the remote pairs. The time slot duration is limited, resulting that at most b_h bytes of data can be downloaded each time slot a user contacts helper h. The different values of the parameters b_h reflect the bandwidth and the average workload heterogeneity of the helpers [1].

Let $v = (V_1, V_2, ..., V_d)$ denote a $d-$step random walk on M, where $V_i \in \{1, ..., n\}$. Observe that a user can encounter more than one times the same helper within the time deadline d. Thus, v is a *multiset*. The probability of accessing the set of helpers in v by a mobile user within d equals to:

$$P(v) = p_{init}(V_1) \prod_{h=1}^{d-1} M_{V_h, V_h+1} \tag{1}$$

Figure 1 depicts the discussed system model.

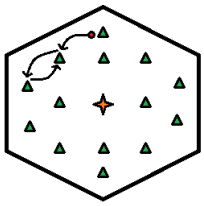


Figure 1. Graphical illustration of the discussed model. MBS lies on the center of the cell. The helpers (triangles) are deployed around MBS. A mobile user (circle) connects to a subset of the helpers within a given time deadline d. Here $d = 4$ and v consists of three distinct helpers, one of which appears twice in v.

In this work, we focus on determining the storage allocation of these $|O|$ files at the n helpers aiming to minimize the fraction of file requests that are served by the main base station of the cell. Let the optimization variable $x_{h,i}$ indicate the fraction of (encoded) data of file O_i stored at helper h to $|O_i|$. We denote with $\mathcal{M}_{n,d}$ the set of all possible multisets consisted of elements in $\{1, 2, ..., n\}$ of size d. We also denote with $\mathcal{S}_v$ the set of distinct helpers that comes from the multiset v by removing the duplicate elements. Let η_v^h denote the number of times that helper h appears in v. Then, the probability of failed file delivery

corresponding to an allocation x equals:

$$P_f(x) = \sum_{v \in \mathcal{M}_{n,d}} P(v) \sum_{i \in O} P_{i/V_1} I_{\{\sum_{h \in \mathcal{S}_v} \sum_{k=1}^{\eta_v^h} u_{h,i}^k < 1\}} (2)$$

where $I_{\{.\}}$ is the indicator function, i.e. $I_{\{S\}} = 1$ iff $S = true$, else $I_{\{S\}} = 0$. $u_{h,i}^k$ denotes the fraction of file i that can be downloaded from helper h when a user contacts h for the k^{th} time. Clearly, *there is no benefit to download again the same data already downloaded from* h *at previous contacts*. Thus, $u_{h,i}^1 = \min\{x_{h,i}, \frac{b_h}{|O_i|}\}$ and $u_{h,i}^k = \min\{x_{h,i} - \sum_{l=1}^{k-1} u_{h,i}^l, \frac{b_h}{|O_i|}\}$, $k = 2, 3, ..., d$.

The problem of performing the storage allocation that minimizes the probability of failed file delivery is the following:

$$\min_x \ P_f(x) \quad (3)$$

$$s.t. \ \sum_{i=1}^{|O|} |O_i| x_{h,i} \leq |C_h|, \forall h = 1, ..., n \quad (4)$$

$$x_{h,i} \in [0, 1], \forall h = 1, ..., n, i = 1, ..., |O| \quad (5)$$

,where inequalities in (4) denote the cache capacity constraints. Inequalities in (5) indicate the non-negativeness of the optimization variables and that it is wasteful to allocate to a cache more than one unit of the same file. The above problem is difficult to solve due to it's non-convex nature and the high number of different multisets of helpers that a user can encounter within the time deadline d. Clearly, there exist n^d such multisets.

IV. Small-scale optimal solution

In this section we show how to solve optimally the discussed storage allocation problem in a small scale. In other words, we focus on scenarios consisted of a small number of helpers (some decades) and a small deadline parameter d. Because of the small value of n^d, we can implicitly enumerate all the possible multisets of helpers encountered by a user within d.

We first define the binary variable $T_i^v(x)$ to denote whether a user can download sufficient amount of data of file i from the helpers encountered according to the random walk v, given the storage allocation x. Thus, $T_i^v(x)$ is defined as follows:

$$T_i^v(x) = \begin{cases} 1, & if \ \sum_{h \in \mathcal{S}_v} \sum_{k=1}^{\eta_v^h} u_{h,i}^k \geq 1 \\ 0, & else \end{cases} \tag{6}$$

Then, we can formulate the discussed problem as a *Mixed Integer Programming (MIP)* problem as below:

$$\min_{x,u,T} \sum_{v \in \mathcal{M}_{n,d}} P(v) \sum_{i \in O} P_{i/V_1}(1 - T_i^v) \tag{7}$$

$$s.t. \ (4) - (5),$$

$$u_{h,i}^k \in [0, \frac{b_h}{|O_i|}], \ k = 1, ..., d, \ \forall h, i \tag{8}$$

$$\sum_{k=1}^{d} u_{h,i}^k \leq x_{h,i}, \ \forall h, i \tag{9}$$

$$\sum_{h \in \mathcal{S}_v} \sum_{k=1}^{\eta_v^h} u_{h,i}^k \geq T_i^v, \forall i, v \tag{10}$$

$$T_i^v \in \{0, 1\}, \forall i, v \tag{11}$$

Inequalities (8)-(9) are added because of the definition of variables u. The number of optimization variables is $|O|(n^d + nd + n)$. It is known that the above problem can be efficiently solved using Branch and Bound algorithms [5] for a small number of optimization variables.

V. Large-scale approximate solution

In this section we establish an approximate solution of the discussed problem for arbitrarily large problem instances by minimizing an appropriate probability bound. A similar approach was used in [2]. We start with the following lemma:

Lemma 1: (Chernoff-Hoeffding probability bound [6])

Let E be an ergodic Markov chain with state space S and stationary distribution π. Let $(V_1, ..., V_t)$ denote a $t-$step random walk on E starting from an initial distribution ϕ on S. For every $i \in \{1, 2, ..., t\}$, let $f_i : S \to [0, 1]$ be a weight function at step i such that the expected weight $E_v[f_i(v)] = \mu$ for all i. Define the total weight of the random walk $(V_1, ..., V_t)$ by $X_t = \sum_{i=1}^{t} f_i(V_i)$. There exists some constant c (which is independent of μ and δ), such that: $Pr[X_t \leq (1-\delta)\mu t] \leq c||\phi||_\pi exp(-\frac{\delta^2 \mu t}{72T})$, for $0 \leq \delta \leq 1$, where T is the mixing time of E, defined as $T = \min\{t : \max_q ||qE^t - \pi||_{TV} \leq \frac{1}{8}\}$, q is an arbitrary initial distribution over E, $||u-v||_{TV} = \max_{A \subseteq V} |\sum_{i \in A} u_i - \sum_{i \in A} w_i|$ and $||u||_\pi = \sqrt{\sum_{x \in S} \frac{u_i^2}{\pi(i)}}$.

Let Y be the random variable indicating the fraction of the requested file that can be downloaded by a user within d, given the storage allocation x. Then, $P_f(x) = P[Y < 1]$. We use lemma 1 to derive an upper bound on the probability $P[Y < 1] \leq P[Y \leq 1]$ for an arbitrary allocation. Y can be interpreted as the total weight $\sum_{i=1}^{d} f_i(V_i)$ of the random walk $(V_1, ..., V_d)$, on an appropriately constructed markov chain E. We construct the Markov chain E as follows:

The state space S of E consists of $1 + |O| \sum_{i=1}^{d} (n^i)$ states. We name one of the states of S as the *root*, indexed by 0. We partition the other states into $|O|$ groups, such that the first group contains the first $\sum_{i=i}^{d} (n^i)$ states, the second group contains the next $\sum_{i=i}^{d} (n^i)$ states etc. The states of these groups combined with the root form an *hierarchy* of $d+1$ levels named as $\{0, 1, ..., d\}$. Root is the only state of level 0. Root is the unique parent of the $|O| * n$ states of level 1, n states for each of the groups. Each state that is included in the level $l \in \{1, ..., d-1\}$ of the group $g \in \{1, ..., |O|\}$ is the unique parent for n states of the level $l+1$ of that group. We define by $child(u, c)$ the c^{th} child-state of state u. Besides, let $\mathcal{G}(u)$ denote the group that contains state u and $\mathcal{P}_u$ denote the set of states that lie on the path between state 0 and state u (including the two endpoints). A state u that belongs to the level $l > 1$ and it is the c^{th} child of another state, represents the connection of a user requesting file $\mathcal{G}(u)$, to helper c at the time slot l. The n first states of level 1 represent

the connection to a node requesting file 1 at time slot 1, the next n states represent the same but requesting file 2 etc. Figure 2 illustrates a simple example of E.

The initial probability ϕ of E is given by:

$$\phi(u) = \begin{cases} P_{\mathcal{G}(u)/c'} P_{init}(c'), & if \ \substack{\exists c \in \{1,...,n*|O|\}: \\ u = child(0,c)} \\ 0, & otherwise \end{cases}$$

where, for ease of presentation, we denoted by $c' = c - (\mathcal{G}(u) - 1)n$. According to $\phi()$, every walk starts at a state of level 1.

The probability of transition from state v to u is given by:

$$E_{v,u} = \begin{cases} 1 - \alpha, & if \ v = u = 0 \\ \alpha P_{\mathcal{G}(u)/c'} P_{init}(c'), & if \ \substack{v=0, \\ \exists c \in \{1,...,n|O|\}: u = child(0,c)} \\ M_{c',e}, & if \ \substack{\exists c \in \{1,...,n|O|\}, e \in \{1,...,n\}: \\ v = child(0,c), u = child(v,e)} \\ M_{c,e}, & if \ \substack{\exists c,e \in \{1,...,n\}, w \in S \setminus 0: \\ v = child(w,c), u = child(v,e)} \\ 1, & if \ |\mathcal{P}_v| = d + 1, u = 0 \\ 0, & otherwise \end{cases}$$

where $a \in (0, 1)$ is a fixed parameter. The transitions from the states of the d^{th} level to state 0 and from state 0 to itself are necessary for E to be ergodic. The weight of a state u is defined as:

$$f_i(u) = \begin{cases} u^1_{c',\mathcal{G}(u)}, & if \ \exists c \in \{1, ..., n|O|\} : u = child(0, c) \\ u^1_{c,\mathcal{G}(u)}, & if \ \substack{\exists c \in \{1,...,n\}, w \in S \setminus 0: \ u = child(w,c), \\ \nexists y \in \mathcal{P}_u \setminus u: \ f_i(y) = u^1_{c,\mathcal{G}(u)}} \\ u^k_{c,\mathcal{G}(u)}, & if \ \substack{\exists c \in \{1,...,n\}, w \in S \setminus 0, \ k \in Z^+: \ u = child(w,c), \\ k = max\{k: \exists y \in \mathcal{P}_u \setminus u: \ f_i(y) = u^{k-1}_{c,\mathcal{G}(u)}\}} \\ 0, & else \end{cases}$$

Observe that according to the definition of the $f_i()$ function, the weight of a state u matches the fraction of the requested file that a user can download during the associated

contact. This weight depends only on the number of the previous contacts of the user to the same helper, as explained in section III.

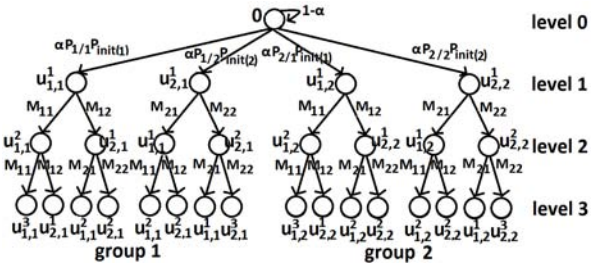


Figure 2. An example of the Markov chain E. The parameters are set as follows: $n = 2$, $d = 3$ and $|O| = 2$. Circles represent the states and arrows the transitions of non zero probability. State labels represent the state weights. Arrow labels represent the probabilities of transition. For ease of presentation we omitted the transitions that start from states of level 3. In reality, for each of these states there is a transition to the root state of probability 1.

By construction of E, every $d-$step walk starts at a state of level 1 and ends at a state of level d, representing a walk of a user over the helpers within d. The requested file is specified by the group that includes these states. $M_{h',h}$ and $P_{i/h}$ may be equal to zero for some values of i, h and h'. In order to ensure ergodicity of E, we exclude the states that are reachable by the root state with zero probability. Let $\mu(u, x)$ be the expected weight of a walk on E as a function of the weights. Set $\delta = 1 - \frac{1}{\mu(u,x)d}$ and require $\mu(u, x)d \geq 1$. Then, lemma 1 yields:

$$P[Y < 1] \leq c||\phi||_\pi exp(-\frac{\mu(u,x)d + \frac{1}{\mu(u,x)d} - 2}{72T}) \qquad (12)$$

,where π is the stationary distribution of E, T is the mixing time of E and c is a constant. By definition, it holds that:

$$\mu(u, x) = \sum_{h=1}^{n} \sum_{k=1}^{d} \sum_{i=1}^{|O|} P[u_{h,i}^k] u_{h,i}^k \quad (13)$$

where $P[u_{h,i}^k]$ denotes the probability of reaching any of the states of E with weight equal to $u_{h,i}^k$ according to π.

Note that π can be easily computed because of the special form of E; for any state $u \in S \setminus 0$ it is $\pi(u) = \pi(0) * E_{0,i_1} * E_{i_1,i_2} * ... * E_{i_l,u}$, where $\mathcal{P}_u = \{0, i_1, ..., i_l, u\}$. Besides, it holds that $\sum_{u \in S} \pi(u) = 1$. Thus, after some computations we find that $\pi(0) = \frac{1}{1+\alpha d}$. $P[u_{h,i}^k]$ can be defined as follows:

$$\frac{P[u_{h,i}^k]}{\pi(0)\alpha} = P_{init}(h)P_{i/h} * \sum_{\substack{l_1+..+l_{k-1} \leq d-1 \\ l_1,..,l_{k-1} \geq 1}} (\prod_{i=1}^{k-1} r_{h,h}(l_i)) +$$

$$+ \sum_{h' \neq h} (P_{init}(h')P_{i/h'} * \sum_{\substack{l_1+..+l_k \leq d-1 \\ l_1,..,l_k \geq 1}} r_{h',h}(l_1) \prod_{i=2}^{k} r_{h,h}(l_i)) \quad (14)$$

,where $r_{i,j}(l)$ denotes the probability that a user connects to the helper j *for the first time* at the l^{th} time slot conditioned on the event that it connects to the helper i at the first time slot. $r_{i,j}(l)$ is defined by the following set of *recursive equations*:

$$r_{i,j}(l) = \sum_{k=1,k \neq j}^{n} r_{i,k}(l-1)M_{k,j} \quad (15)$$

$$r_{i,j}(1) = M_{i,j} \quad (16)$$

Equations (15)-(16) are very similar to the Chapman-Kolmogorov equations. However, Chapman-Kolmogorov equations do not require $k \neq j$ over the above summation. Simply speeking, equation (14) specifies that a user can initially be at helper h and then encounters h again $k-1$ times, or the user can initially be at a different helper h' and then encounters h k times before the deadline d expires.

In order to derive an approximate solution of the problem described in (3)-(5) we minimize the upper bound in (12). Lemma 2 shows that minimizing the aforementioned bound is equivalent to maximizing the expected weight $\mu(u, x)$.

Lemma 2: Let $g(u,x) = c||\phi||_\pi \exp(-\frac{\mu(u,x)d+\frac{1}{\mu(u,x)d}-2}{72T})$.

$$\text{Then, } \underset{(u,x)\in\mathcal{A}}{\operatorname{argmin}}\ g(u,x) = \underset{(u,x)\in\mathcal{A}}{\operatorname{argmax}}\ \mu(u,x)$$

where $\mathcal{A} = \{(u,x) \in \mathbb{R}^{n\times|O|\times d} \times \mathbb{R}^{n\times|O|} : (4)-(5),(8)-(9) \text{ are satisfied}\}$.

Proof: Let $(u^*,x^*) = \underset{(u,x)\in\mathcal{A}}{\operatorname{argmin}}\ g(u,x)$. Then, $g(u^*,x^*) \leq g(u,x),\ \forall(u,x) \in \mathcal{A}$. Dividing by $c||\phi||_\pi$ and then taking the logarithm on both sides preserves the inequality as $c>0$, $||\phi||_\pi > 0$ and $log(x)$ is strictly increasing. Thus, we have:

$$-\frac{\mu(u^*,x^*)d+\frac{1}{\mu(u^*,x^*)d}-2}{72T} \leq -\frac{\mu(u,x)d+\frac{1}{\mu(u,x)d}-2}{72T}$$

Dividing by $-72T \leq 0$ and then adding 2 on both sides yields:

$$\mu(u^*,x^*)d + \frac{1}{\mu(u^*,x^*)d} \geq \mu(u,x)d + \frac{1}{\mu(u,x)d}$$

However, it holds that: $\mu(u,x)d \geq 1$, resulting that: $\mu(u^*,x^*)d \geq \mu(u,x)d,\ \forall(u,x) \in \mathcal{A}$

■

The optimization problem becomes as follows:

$$\max_{u,x}\ \mu(u,x) \tag{17}$$
$$s.t.\ (4),(5),(8),(9)$$

By the structure of $\mu(u,x)$, that is defined in (13), and the above problem constraints, we observe that the storage allocation decisions at a helper do not affect the storage allocation decisions at the other helpers. Thus, we can *decompose* the problem to n independent *linear programming* subproblems, one for each helper, and solve them in a *distributed* manner. It is known that he numerical solution of a linear programming problem can be efficiently attained using the simplex method. However, as we show below each of these subproblems falls into a class of tractable problems, with *known solution structure*, alleviating the need for applying the simplex method.

Fractional knapsack problem asks for placing fractions of materials of different values and weights in a knapsack of limited capacity in order to maximize the aggregate value of the materials placed in it. The storage allocation at a helper h subproblem can be translated to a *restricted version of the fractional knapsack problem* in which there exist $|O| * d$ materials, one material for each variable $u_{h,i}^k$, $i = 1, ..., |O|$, $k = 1, ..., d$ and a knapsack of capacity $|C_h|$. The value of the material corresponding to $u_{h,i}^k$ is $P[u_{h,i}^k]$ and it's weight is $|O_i|$. The material placement must also satisfy the following two constraints: 1) We are restricted to place in the knapsack at most a $\frac{b_h}{|O_i|}$ fraction of the material corresponding to variable $u_{h,i}^k$. 2) The sum of the fractions of the materials corresponding to variables $u_{h,i}^k$, $k = 1, ..., d$, placed in the knapsack must not be greater than 1. Using the exact same arguments used in [8], we can prove that the optimal solution of this knapsack-type problem can be attained by the following *greedy* algorithm:

> Sort the materials in decreasing order of value per unit of weight. Then insert them into the knapsack, starting with as large amount as possible of the first material, without violating any of the above two constraints, until there is no longer space in the knapsack for more.

The solution of this knapsack-type problem, attained by the greedy algorithm, translates to a solution to the original storage allocation at helper h subproblem such that $u_{h,i}^k$ variable takes the value equal to the fraction of the associated material placed in the knapsack and $x_{h,i}$ takes the value equal to $\sum_{k=1}^{d} u_{h,i}^k$. The solution is independent of the values of α and $\pi(0)$. The complexity of the greedy algorithm comes mainly to the sorting of the values of the materials. Quicksort is the fastest sorting algorithm, resulting *complexity* of $|O|d \log(|O|d)$.

VI. Performance evaluation

In this section we present the numerical experiments that we have conducted to evaluate the performance of the proposed algorithms. The algorithms have been applied to a cellular network consisted of a single cell and $n = 623$ helpers. The geographical position of the

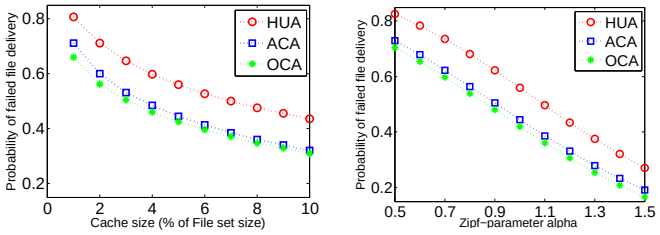
helpers as well as the mobility pattern of the users were acquired by the recording of wireless traffic logs included in the CRAWDAD data set [9]. The set contained 13888 logs, one for each mobile user, including details about the encountered helpers as well as the time that each encounter happened. According to our model, location transitions of the users happen in a time slotted fashion. We set the time slot duration to be equal to 100 seconds. Based on the data set, we set $P_{init}(h)$ to be equal to the frequency of time slots at which a user starts it's walk from helper h. Similarly, $M_{i,j}$ takes as value the frequency of time slots at which a user encounters sequentially within the same time slot the helpers i and j. If the user does not encounter any other helpers by the end of the time slot, then we assume that it encounters again the same helper increasing the value of $M_{i,i}$.

In all simulations, we assume a collection of $|O| = 100$ files each one of size 30 MB. We use a Zipf-Mandelbrot model to formulate the files request pattern with a shape parameter $alpha$ and a shift parameter $q = 10$. At each time slot at most $b_h = 15$ MB of the requested data file can be downloaded by the encountered helper h. A user is satisfied when the requested file is delivered to it within a time deadline that is equal to 300 seconds, i.e. $d = 3$ time slots. Such a time deadline is reasonable for a playback time for a typical video file.

Throughout, we compare the probability of failed file delivery achieved by three algorithms:

1) Heuristic Uncoded Algorithm (HUA): the standard mode of operation currently in use in most storage systems. Each helper stores the most popular files that fit in it's cache independently from the others.
2) Approximation Coded Algorithm (ACA): The solution of the knapsack-type problem described in section V.
3) Optimal Coded Algorithm (OCA): The solution of the MIP problem described in section IV.

Figure 3(a) shows the results as a function of the cache size of the helpers for $alpha = 1$.



a) Impact of cache sizes b) Impact of Zipf-parameter

Figure 3. Probability of failed file delivery as a function of a) the cache sizes and b) the Zipf-parameter $alpha$.

The cache size of each helper is varied from 1% to 10% of the file set size. As expected, increasing the cache size of each helper reduces the probability of failed file delivery. Figure 3(b) shows the results as a function of the parameter $alpha$. The cache size of each helper was set to 5% of the entire file set size. We can see from the graph that the probability of failed file delivery decreases with increasing values of $alpha$, reflecting the well known fact that caching effectiveness improves as the popularity distribution gets steeper. In general, we observe that OCA is strictly better than ACA, which in turn is better than the HUA. The performance achieved by the OCA and ACA are very close. This indicates that helpers can independently take local storage decisions, and still have a significant gain. In summary, our algorithms perform $20-50\%$ better than the conventional file placement scheme.

VII. Conclusion

In this chapter we introduced a new storage allocation scheme for offloading traffic from the cellular network. Our work builds upon a recent network architecture [1], with concerns on user mobility and limited contact duration time. Our main contribution is a distributed light-weight storage allocation algorithm. We used a real trace of user movements and demonstrated significant performance gains compared to conventional schemes. Our algorithms can be easily extended to handle the case that the coverage areas of the helpers are overlapping.

REFERENCES

[1] N. Golrezaei, K. Shanmugam, A. Dimakis, A. Molisch, G. Caire, "FemtoCaching: Wireless Video Content Delivery through Distributed Caching Helpers", INFOCOM, 2012.

[2] V. Ntranos, G. Caire, A. Dimakis, "Allocations for Heterogenous Distributed Storage", ISIT, 2012.

[3] D. Leong, A. Dimakis, T. Ho, "Distributed Storage Allocations for Optimal Delay", ISIT, 2011.

[4] D. Leong, A. Dimakis, and T. Ho, "Distributed storage allocations", CoRR, abs/1011.5287, 2010.

[5] X. Zhuo, Q. Li, W. Gao, G. Cao, Y. Dai, "Contact Duration Aware Data Replication in Delay Tolerant Networks", ICNP, 2011

[6] K. Chung, H. Lam, Z. Liu, M. Mitzenmacher, "Chernoff-Hoeffding Bounds for Markov Chains: Generalized and Simplified", STACS, 2012.

[7] C. Geyer, "Practical markov chain monte carlo", Statistical Science, pp. 473483, 1992.

[8] G. Dantzig, "Discrete-Variable Extremum Problems", Operations Research Vol. 5, No. 2, pp. 266288, 1957.

[9] http://crawdad.cs.dartmouth.edu/meta.php?name=dartmouth/campus

[10] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update", Feb. 2012.

Chapter 2:

Approximation Caching and Routing Algorithms for Massive Mobile Data Delivery

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 has been proposed in order to decrease the capacity -and hence the cost- of the backhaul links that connect these base stations with the core network. However, deriving the optimal caching policy remains a challenging open problem especially if one considers realistic parameters such as the bandwidth limitation of the base stations. The latter constraint is particularly important for cases when users requests are massive. We consider such a scenario and formulate the joint routing and caching problem aiming to maximize the fraction of content requests served by the deployed base stations. This is an NP-hard problem and hence we cannot obtain exact optimal solution. Thus, we present a novel approximation framework based on a reduction to a well known variant of the facility location problem. This allows us to exploit the rich literature in facility location problems, in order to establish bounded approximation algorithms for our problem.

I. Introduction

In order to cope with the mobile data traffic explosion, network operators (MNO) deploy small cell base stations which operate in conjunction with the macro-cellular base stations. This architecture increases coverage density, optimizes the reuse of spectrum by exploiting

spatial diversity (cell splitting) and benefits 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 of the MNO. This is currently one of the major operation cost component for mobile network operators [1].

Distributed caching architectures have been recently proposed [2] and evaluated [1] with the goal to minimize peak traffic - and subsequently the cost - of these backhaul links. The idea is to cache in advance popular content items at the small-cell base stations so as to reduce, especially during peak traffic hours, the requests that are routed over the backhaul links to the core network. Given the vast set of the content items, the challenge is to find the optimal caching policy, i.e., to decide which items should be cached at each base station, so as to maximize the portion of user requests that are satisfied locally. Unfortunately though, this has been proved to be an NP-hard problem [2].

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 this case, mobile data delivery will be constrained by the limited transmission capacity of the base stations. Clearly, in order 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 transmit it. Prior works assume that transmission capacity is rarely the bottleneck for the caching base stations. Clearly, this is not a realistic assumption for massive content demand scenarios.

In this work, we consider the realistic scenario of massive content delivery through small-cell base stations 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 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 request

to route to each base station. Even more interestingly, the routing and caching policy must be jointly optimized. We provide an analytical framework for this *joint caching and routing* (JCR) problem the solution of which maximizes the content requests that are satisfied by the small-cell base stations.

Clearly, this problem is equally important and challenging to solve. Namely, the already NP-hard caching problem [2] is further compounded due to the additional bandwidth constraints of small-cell base stations. Besides, our model additionally takes into account the *heterogeneity of the base stations* which may have different cache sizes and transmission capacity and the *variation of request patterns of the users* which may ask for different content with different probability/rate/density. Finally, we assume that user content *requests are unsplittable*, i.e., each request is satisfied by one base station (no content splitting in different base stations is assumed) which is the most challenging scenario in caching problems.

To overcome these difficulties we propose a novel method for mapping the JCR problem to a well-investigated variant of the facility location problem. This way we can use, after proper modifications, the class of algorithms that have been derived for this problem. Specifically, we present a set of novel *bounded approximation algorithms* for the JCR problem. To the best of the authors knowledge, this is the first work that provides a bounded approximation solution for the joint content placement and routing problem in a wireless caching network. The contribution of this work is thus three-fold:

- We consider a realistic cellular model, which includes important features such as the heterogeneity of network resources of the base stations and different content request patterns for the users. We formulate the joint problem of caching and routing aiming to maximize the requests that are served by the small-cell base stations.
- We reduce the above problem to the *Unsplittable Hard-Capacitated Metric Facility Location Problem (UHCMFL)* [12]. This result reveals the potential for exploiting the rich literature in facility location problem solutions to answer open questions for the

problem under consideration.

- We present an approximation framework based on the aforementioned reduction. Particularly, we show that every approximation algorithm for *UHCMFL* can be translated to an algorithm with a bounded approximation ratio for our problem.

The remainder of the chapter is organized as follows. Section II reviews our contribution compared to related works. Sec. III describes the system model and the assumptions, and introduces formally the problem. In Section IV, we reduce the problem to the *UHCMFL* problem. Sec. V presents an approximation framework based on the above reduction. A summary concludes the chapter in Section VI.

II. Related work

Cooperative caching in a wireline setting has long been investigated. Korupolu et al. [4] 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. [5] developed approximation caching algorithms aiming to minimize the bandwidth cost. In our prior work [3], 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 the realistic case that hard bandwidth constraints limit the transmission of the caching base stations.

The potential of request routing for enhancing content delivery has also been investigated. Betkas et al. [6] studied the joint caching and request routing problem in CDNs. Besides, the same problem is studied for content delivery over IPTV wireline networks in [7]. Both 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 evaluated by simulations only. In contrast, our algorithms provide a *bounded* approximation ratio and

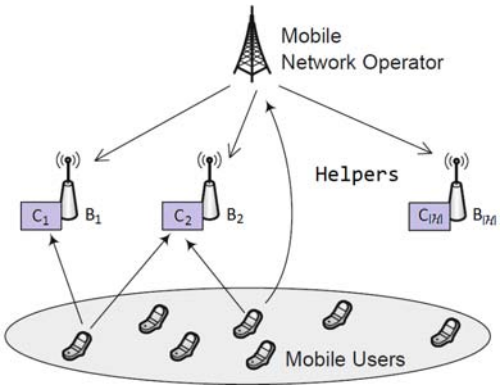


Fig. 1. Mobile users are randomly distributed in the coverage regions of the helpers. Each helper i has certain storage and bandwidth capacity of C_i and B_i units respectively.

require *bounded* computational complexity.

Besides, our work considers a wireless setting and builds upon a novel architecture [2]. Authors in [2] assume that the transmission bandwidth of the base stations in the wireless network suffices to serve all the content requests of the users. Thus, they focus on deriving the caching policy and simply associate each content item request to the nearest base station that has a copy of it. In contrast, we consider the realistic case that the base stations are *congestible*, which highlights the need for coupled request routing and caching decisions in order to fully utilize the transmission capabilities of the base stations.

III. System model and problem formulation

We consider a single macrocell in which the mobile network operator (MNO) serves the content requests of its subscribers, the mobile users. A set of small base stations are also deployed within the cell, working in conjunction with the main base station. This two-layer architecture is depicted in Fig. 1. We assume that time is slotted and we study the system for one time period T. Following the terminology in [2], we name these base stations as *helpers* and denote with $\mathcal{H}$ the associated set. Each helper $h \in \mathcal{H}$ has a certain transmission capacity, i.e. it can deliver B_h data bytes within period T[1]. Besides, each helper h is equipped with a cache of size C_h bytes.

[1]In different scenarios,B_h can capture other cost-related metrics such as the transmission energy cost for the helpers.

Let the set $\mathcal{O}$ indicate a static collection of 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 [2], [5]. Let $\mathcal{K}$ denote the set of mobile users within the cell in the considered time interval[2]. Users are heterogeneous since they may have different content demands. We denote with $\lambda_{ki} \in \mathcal{Z}^+$ the estimated number of requests for file $i \in \mathcal{O}$ generated by user $k \in \mathcal{K}$ within T. User requests may change over consecutive time periods but are considered fixed within each period. This is a realistic assumption as the popularity distribution of the files changes slowly [2]. The coverage areas of the helpers are *overlapping* in general. We denote $\mathcal{N}(k) \subseteq \mathcal{H}$ the set of helpers that are in communication range with user k. Then, a request generated by k can be satisfied by any of the helpers in $\mathcal{N}(k)$ that owns a copy of the requested item. We assume that unsatisfied requests are routed to the main base station. This exactly is the quantity we need to minimize.

Before we proceed, let us provide a simple example which highlights how the limited helper capacity impacts the caching policy.

Motivating Example. Consider the system depicted in Fig. 2 with two helpers ($h1$ and h_2) and three users (u_1, u_2 and u_3). The circles represent the coverage areas of the helpers. There are also two equal-sized files, named as f_1 and f_2. Each helper can cache at most one file due to its limited storage capacity. Also, because of the bandwidth limitations, h_1 can serve at most 5 requests and h_2 can serve at most 10 requests. User u_1 requests f_1 1 time, u_2 requests f_1 2 times and u_3 requests f_2 10 times. The optimal strategy (which maximizes the requests satisfied by h_1 and h_2) is to cache f_1 to h_1 and f_2 to h_2. Then, h_1 serves the request for f_1 generated by u_1, and h_2 serves all the requests for f_2 generated by u_3. Hence, only 2 requests must by served by the main base station.

However, if we omit the transmission capacity constraints, then the *optimal caching policy changes*; it places f_2 to h_1 and f_1 to h_2. Then, h_1 handles all the requests of u_3,

[2]A user $k \in \mathcal{K}$ may represent many users that are on the same location in the cell.

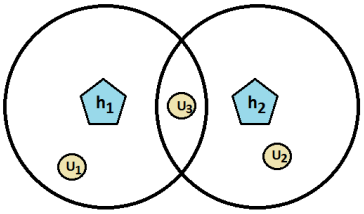


Fig. 2. An example with 2 helpers (h_1 and h_2) and 3 users (u_1, u_2, u_3). The circles denote the transmission range of each helper.

and h_2 handles all the requests of u_2, letting only the request generated by u_1 to be routed to the main base station. Nevertheless, in practice, h_1 will serve only half of incoming requests (due to limited capacity) and redirect the rest to the main base station. Hence, the main base station needs to serve 6 requests in total. This example demonstrated that *ignoring the helpers' bandwidth capacities lead the system to inefficient operating points*, for the case of massive content requests where the capacity limits of the helpers are reached.

Problem Statement. Let us introduce the integer decision variable $x_{hi} = \{0, 1\}$ which indicates whether file $i \in \mathcal{O}$ is placed at the cache of helper $h \in \mathcal{H}$ or not. We also define the respective *caching policy* matrix $\boldsymbol{x} = (x_{hi} : h \in \mathcal{H}, i \in \mathcal{O})$. Besides, let the integer decision variable $y_{hi}^k \in \mathcal{Z}^+$ indicate the number of requests for file i generated by user k that are routed to helper h. Also, y_{0i}^k denotes the number of requests for file i generated by user $k \in \mathcal{K}$ that are routed to the main base station of the cell. We define the *routing policy* matrix $\boldsymbol{y} = (y_{hi}^k : h \in \mathcal{H} \cup \{0\}, i \in \mathcal{O}, k \in \mathcal{K})$. Observe that each one of the λ_{ki} requests for file i generated by user k must be satisfied by exactly one helper, $\forall i \in \mathcal{O}, k \in \mathcal{K}$. This 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 helpers [6], [7].

The problem of devising the routing and caching policy which minimizes the load of the main base station may then be formulated as follows:

$$\min_{\boldsymbol{x},\boldsymbol{y}} \quad \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}} y_{0i}^{k}s \tag{1}$$

$$s.t. \quad \sum_{i\in\mathcal{O}} x_{hi}s \leq C_h,\ \forall h \in \mathcal{H} \tag{2}$$

$$\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}} y_{hi}^{k}s \leq B_h,\ \forall h \in \mathcal{H} \tag{3}$$

$$y_{hi}^{k} \leq x_{hi}\lambda_{ki},\ \forall i \in \mathcal{O}, k \in \mathcal{K}, h \in \mathcal{H} \tag{4}$$

$$y_{hi}^{k} = 0,\ \forall i \in \mathcal{O},\ k \in \mathcal{K}, h \in \mathcal{H} \setminus \mathcal{N}(k) \tag{5}$$

$$\sum_{h\in\mathcal{H}\cup\{0\}} y_{hi}^{k} = \lambda_{ki},\ \forall i \in \mathcal{O}, k \in \mathcal{K} \tag{6}$$

$$x_{hi} \in \{0,1\},\ \forall h \in \mathcal{H},\ i \in \mathcal{O} \tag{7}$$

$$y_{hi}^{k} \in \mathcal{Z}^{+},\ \forall h \in \mathcal{H}\cup\{0\}, i \in \mathcal{O},\ k \in \mathcal{K} \tag{8}$$

Inequalities (2) and (3) denote the cache capacity constraints and the bandwidth limitations of the helpers respectively. Also, (4) indicates that helpers can not serve requests for files that are not in their caches. Constraints (5) denote that helpers can not serve requests generated by users located out of their coverage areas. Equalities (6) dictate that the system must serve all the requests (inelastic demands). Finally, (7)-(8) indicate the discrete nature of the optimization variables. We call the above the JCR problem.

Clearly, the above problem is NP-hard, as it is a generalization of the *Helper Decision Problem*, described in [2], by incorporating hard bandwidth capacity constraints at the helpers.

IV. Reduction to a variant of the Facility Location problem

In this section, we describe a polynomial time reduction of the JCR problem to a variant of the facility location problem [12]. The latter is formally defined as follows:

Definition 1: Unsplittable Hard-Capacitated Metric Facility Location Problem (UHCMFL): A set of facilities $\mathcal{A}$ and clients $\mathcal{B}$ are given. Let d_i denote the demand of client i. Besides, let f_j and S_j denote the opening cost and the capacity of facility j respectively. Each client

needs to assign it's entire demand to a single open facility (*unsplittable*). S_j limits the total sum of demands served by facility j (*hard capacitated*). We denote by c_{ij} 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 $c_{ij} + c_{jk} \geq c_{ik}$, $\forall i, j, k \in \mathcal{A} \cup \mathcal{B}$. The problem asks for specifying which subset of facilities $\mathcal{A}^* \subseteq \mathcal{A}$ to open and to determine which clients each one of them will serve (denoted by a function $\pi : \mathcal{B} \rightarrow \mathcal{A}^*$) such that the overall cost $\sum_{j \in A^*}(f_j) + \sum_{i \in \mathcal{B}}(d_i c_{i\,\pi(i)})$ is minimized, without violating the capacity inequalities $\sum_{\{i \in \mathcal{B}: \pi(i)=j\}} d_i \leq S_j$, $\forall j \in \mathcal{A}$.

The connection between $UHCMFL$ and the JCR 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 [12]. Our model substantially differs from these works, as it considers the practical case that *multiple content objects* exist, while the cache size and the bandwidth capacity of the helpers 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: Given a unit time oracle for $UHCMFL$ problem, we can solve the JCR problem in polynomial time.

We describe in detail this reduction and prove it's 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 JCR problem to an instance of the $UHCMFL$ problem. Let F_{JCR} be that instance of the $UHCMFL$ problem. Then, F_{JCR} is constructed as follows:

The set of facilities $\mathcal{A}$ consists of: (i) a facility named a_{BS} and (ii) a facility named a_{hi} for every helper h and every file i. The set of clients $\mathcal{B}$ consists of the following: (i) λ_{ki} clients, denoted as $b_{ki1}, b_{ki2} \ldots, b_{ki\lambda_{ki}}$, $\forall k \in \mathcal{K}$ and $\forall i \in \mathcal{O}$, (ii) $|\mathcal{O}| - \lfloor \frac{C_h}{s} \rfloor$ clients, named

as b'_{h1}, b'_{h2} etc, $\forall h \in \mathcal{H}$ and (iii) $(\lfloor \frac{C_h}{s} \rfloor - 1)\lfloor \frac{B_h}{s} \rfloor$ clients, named as b''_{h1}, b''_{h2} etc, $\forall h \in \mathcal{H}$. The capacity of the facility a_{BS} is set to $+\infty$ and to $\frac{B_h}{s}$ for each a_{hi}, $\forall h,\, i$. The demand of a client of the form b'_{hi} is equal to $\frac{B_h}{s}$, $\forall h,\, i$. Each of the remaining clients has demand equal to 1. Let c be a small positive constant. Then, the unit serving cost for each pair of a facility and a client is specified as follows: (i) each pair of the form $(a_{BS},\, b_{kij})$, $\forall k, i, j$, has cost equal to $1 + \frac{1}{2} + c$, (ii) each pair of the form $(a_{hi},\, b_{kij})$, such that $h \in N(k)$ and $j \in \{1, ..., \lambda_{ki}\}$, has cost equal to $\frac{1}{2} + c$, (iii) each pair of the form $(a_{hi},\, b'_{hj})$, $\forall h, i, j$, has cost equal to $\frac{1}{2} + c$, (iv) each pair of the form $(a_{hi},\, b''_{hj})$, $\forall h, i, j$, has cost equal to $\frac{1}{2} + c$. 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* [12]. Finally, the facility opening cost is set to zero for every facility.

Roughly speaking, the facility a_{BS} represents the main base station of the cell and facilities a_{hi}, $\forall i$, the helper h. Clients of the type b_{kij}, $\forall k, i, j$ represent the actual user requests, while $b'_{h,j}$, $\forall j$ and $b''_{h,j}$, $\forall j$ denote *virtual* user requests that are necessary to preserve the cache capacity and bandwidth constraints of the helpers, as it will become clear in the following subsection.

Each solution for the F_{JCR} problem can be mapped to a solution for the JCR problem according to the following set of rules:

- Rule 1: For each facility a_{hi} *not* serving any client of the form b'_{hj}, $\forall j$, place file i to the cache of helper h.
- Rule 2: For each facility of the form a_{hi} serving a client of the form b_{kij}, $\forall h, i, k, j$ route a request of user k for file i to helper h.
- Rule 3: The remaining requests are routed to the main base station of the cell.

Figure 3 depicts the reduction for the toy-system of figure 2. Here, we set the system values as follows: $|\mathcal{O}| = 4$, $s = 1$, $C_1 = C_2 = 2$ and $B_1 = B_2 = 2$. Each of the two first users requests every file once. User u_3 performs two requests for the first file. Squares

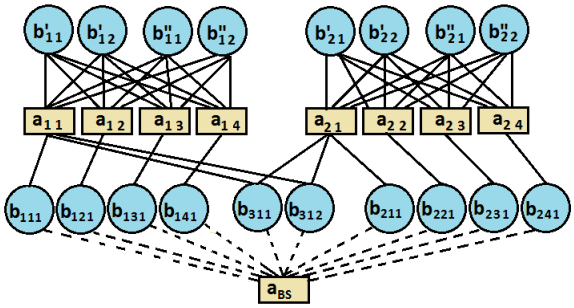


Fig. 3. The reduction of network of Fig. 2 to a respective $UHCMFL$ problem.

represent the facilities and circles the clients. Solid lines unite clients to facilities with cost $\frac{1}{2}+c$. Dashed lines mean that the corresponding cost is $1+\frac{1}{2}+c$. The cost value of each of the remaining pairs is equal to the cost of the shortest path that unite this pair. For example, the cost between the client b'_{11} and facility a_{21} is $\frac{3}{2}+3c$. The demand of each client is 1, except for the clients named as $b'_{hi}, \forall h, i$, whose demand is 2. The capacity of each facility is 2, except for the a_{BS} facility, whose capacity is $+\infty$.

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_{JCR}. Then, we have:

Lemma 1: For every feasible solution of the JCR problem with total cost C, there is a feasible solution to F_{JCR} with total cost $C+D(c+\frac{1}{2})$.

Proof: We construct the solution to F_{JCR} as follows: (i) We open all the facilities at zero cost. (ii) For each file i *not* cached at helper h, we assign the demand of one client of type b'_{hj}, $j \in \{1,...,|\mathcal{O}|-\lfloor\frac{C_h}{s}\rfloor\}$ to the facility a_{hi}. (iii) For each request generated by a user k for a file i served by a helper h, we assign the demand of one client of type b_{kij}, $j \in \{1,...,\lambda_{ki}\}$ to the facility a_{hi}. (iv) The demand of a client of type b''_{hj}, $j \in \{1,...,(\lfloor\frac{C_h}{s}\rfloor-1)(\lfloor\frac{B_h}{s}\rfloor)\}$, is randomly assigned to one of the facilities of the form a_{hi}, $\forall i$, without violating their capacity constraints. (v) For each client b_{kij} that has not been covered yet, we assign it's demand to the facility a_{BS}. Thus, every unit of demand of the clients was assigned to a facility. An assignment to the facility a_{BS} incurs a per

unit cost equal to $1 + \frac{1}{2} + c$, while all the other assignments incur a per unit cost equal to $c + \frac{1}{2}$. By construction of the graph, the total demand assigned to a_{BS} is equal to the number of requests that are routed to the main base station of the cell. Thus, the solution has cost equal to $C + D(\frac{1}{2} + c)$. ■

Lemma 2: For every minimum cost solution of the F_{JCR} instance with total cost C, there is a feasible solution to the JCR problem with total cost $C - D(\frac{1}{2} + c)$.

Proof: We construct the solution to JCR problem as follows:

(i) *For each facility* a_{hi} not *serving any client of type* b'_{hj}, $\forall j$, *place file* i *to the cache of helper* h (Rule 1). Observe that each client b'_{hj}, $\forall j$, must be assigned to a facility of the form a_{hi}, $\forall i$, at per unit cost $\frac{1}{2} + c$. This is because each of the other choices incurs at least $1 + \frac{1}{2} + 3c$ per unit cost. Thus, the extra cost paid is at least $1 + 2c$. From the other hand, each client b_{kij}, $\forall k, i, j$, can always be assigned to the facility a_{BS} at per unit cost $1 + \frac{1}{2} + c$. This means that, the potential gain for assigning it to a facility of the form a_{hi}, $\forall i$, at cost $\frac{1}{2} + c$, 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_{hi}, $\forall i$. There are $|\mathcal{O}| - \lfloor \frac{C_h}{s} \rfloor$ such clients. Thus, these clients fully occupy the capacity of $|\mathcal{O}| - \lfloor \frac{C_h}{s} \rfloor$ of these facilities. Consequently, exactly $\lfloor \frac{C_h}{s} \rfloor$ of the above facilities will remain *uncovered* corresponding to the files placed at the cache of helper h.

(ii) *For each facility of the form* a_{hi} *serving a client of the form* b_{kij}, $\forall h, i, k, j$, *route a request of user* k *for file* i *to helper* h (Rule 2). Observe that each of the clients of type b''_{hj}, $\forall j$, must be assigned to one of the $\lfloor \frac{C_h}{s} \rfloor$ *uncovered* facilities of the form a_{hi}, $\forall i$, similarly to the above case. The capacity of each of these facilities is equal to $\frac{B_h}{s}$. There exist $(\lfloor \frac{C_h}{s} \rfloor - 1)\lfloor \frac{B_h}{s} \rfloor$ such clients, each of them with demand equal to 1. Thus, the remaining capacity suffices for serving at most $\frac{B_h}{s}$ units of demand of the clients b_{kij}, $\forall k, i, j$. By construction, a client b_{kij} can be served by a facility a_{hi} with cost equal to $\frac{1}{2} + c$ iff $h \in N(k)$. The cost for serving b_{kij} by a_{hi}, $\forall h \notin N(k)$ is more than the serving

cost by a_{BS}. Thus, at most $\frac{B_h}{s}$ requests generated by users in the coverage area of a helper h will be routed to $h, \forall h$. The remaining $C - D(\frac{1}{2} + c)$ requests will be routed to the main base station of the cell (Rule 3). ■

V. Approximation Algorithms

In this section, we present an approximation framework for the JCR problem based on the reduction described in Sec. IV. We first discuss the existing approximation algorithms for the facility location problem and describe ways to extend them to tackle our problem. Then, we derive improved approximation ratios for the special case of equal-capacity helpers. This is an important case because in general helpers will be of the same type and hence, they will have equal transmission capacity.

It is NP-hard even to approximate the solution of $UHCMFL$ problem within a bounded factor. Hence, previous works [8]-[13] focused on obtaining *bicriteria* approximation algorithms. Formally, an $(\alpha, \beta)-$ bicriteria 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 it's capacity. Similarly, we can define an $(\alpha, \beta)-$bicriteria approximation algorithm for the JCR problem, such that it's solution violates the bandwidth capacities of the helpers 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 $\alpha-$ approximation algorithm. Table I summarizes the main results in this area. Notice that different results are obtained for the case the facicilities have equal capacities (uniform case). Parameter $\epsilon > 0$ is arbitrarily small and $n = |\mathcal{A}| + |\mathcal{B}|$. A Quasi- polynomial time algorithm $(QP.)$ runs slower than polynomial time $(P.)$, yet faster than exponential time [12].

Although JCR 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 bicriteria bound changes when translating the solution to handle the JCR case. Let $c' = \frac{D(\frac{1}{2}+c)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki}) - \sum_{h\in\mathcal{H}}(B_h/s)}$. Then, we have:

Theorem 2: For any $(\alpha, \beta)-$ bicriteria approximation algorithm for the $UHCMFL$ problem there is an $(\alpha+(\alpha-1)c', (\beta-1)|\mathcal{O}|+1)-$approximation algorithm for the JCR problem, requiring the same computational complexity.

Proof: The overall traffic routed to a helper $h \in \mathcal{H}$ by the real users is $\beta|\mathcal{O}|\frac{B_h}{s}-(|\mathcal{O}|-\lfloor\frac{C_h}{s}\rfloor)\frac{B_h}{s}-(\lfloor\frac{C_h}{s}\rfloor-1)\lfloor\frac{B_h}{s}\rfloor$ in the worst case. That is because, each helper corresponds to $|\mathcal{O}|$ facilities, each of which has capacity equal to the capacity of the helper. The virtual clients of the type b'_{hj} and b''_{hj}, $\forall j$, must be served by a_{hi} in any case, as explained in lemma 2. That is the reason that we subtracted the traffic sent to them in the above expression. However, in reality, only $\frac{B_h}{s}$ amount of data can be transmitted by each helper. The fraction of the two values, after some computations, can be written as $(\beta-1)|O|+1$.

Besides, let opt and $approx$ be the optimal and an approximate solution of the JCR problem respectively. By lemma 1, it holds that: $approx + D(\frac{1}{2}+c) \leq \alpha(opt + D(\frac{1}{2}+c))$ or equivalently: $approx \leq (\alpha + \frac{D(\frac{1}{2}+c)(\alpha-1)}{opt})opt$. Finally, $opt \geq \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki}) - \sum_{h\in\mathcal{H}}(B_h/s)$. ■

Theorem 2 combined with the two first Algorithms in Table I, provides two bicriteria approximation algorithms for the JCR problem. Corollary 1 describes this result:

Corollary 1: There exist a *polynomial time* $(11+10c', |\mathcal{O}|+1)$-approximation algorithm and a *quasi-polynomial time* $(\log(n)+(\log(n)-1)c', \epsilon|\mathcal{O}|+1)$-approximation algorithm, $\forall\epsilon>0$, for the JCR problem, where $n=|\mathcal{A}|+|\mathcal{B}|$.

We can use the above bicriteria solutions to perform the cache placement and the routing of user requests in our problem. However, as the bandwidth capacities of the helpers may be violated by the above factors, the operator may need to endow the base stations with additional capacity, in order to ensure the described approximation ratio. However, in many cases, the operator is unwilling to perform additional investments. Thus, the additional requests that reach a helper will be rerouted to the main base station of the cell, increasing it's load. Theorem 3 characterizes the worst case scenario in terms of the quality of the resulted solution.

TABLE I
BICRITERIA BOUNDS FOR THE $UHCMFL$ PROBLEM.

Algorithm	Case	Bound	Complexity	Reference
1	General	$(11, 2)$	P.	[13],[10],[8]
2	General	$(\log n, 1+\epsilon)$	QP.	[11]
3	Uniform	$(9, 4)$	P.	[8]
4	Uniform	$(O(1), 2)$	P.	[9]
5	Uniform	$(5, 2)$	P.	[13],[11],[8]
6	Uniform	$(\log n, 1+\epsilon)$	P.	[12]
7	Uniform	$(10.173, \frac{3}{2})$	P.	[13]
8	Uniform	$(30.432, \frac{4}{3})$	P.	[13]

Theorem 3: For any $(\alpha, \beta)-$ bicriteria approximation algorithm for the $UHCMFL$ problem there is an $(\alpha + (\alpha-1)c')\frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-\sum_{h\in\mathcal{H}}(B_h/s)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-((\beta-1)|\mathcal{O}|+1)\sum_{h\in\mathcal{H}}(B_h/s)}-$approximation algorithm for the JCR problem, requiring the same computational complexity.

Proof: Let H_β be the number of requests routed to the helpers and M_β be the number of requests routed to the main base station of the cell, 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 helpers will be rerouted to the main base station, as the helpers can not serve them. Let H be the number of requests served by the helpers and M the number of requests served by the main base station, after this rerouting. By definition, $H_\beta + M_\beta = H + M = \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})$. Based on Theorem 2, it holds that: $H_\beta = H((\beta-1)|\mathcal{O}|+1)$. Besides, $H = \sum_{h\in\mathcal{H}}\lfloor\frac{B_h}{s}\rfloor$, as each helper will only serve as many requests as it's capacity allows, after the rerouting of requests. Let $M = mM_\beta$, where $m \in \mathcal{R}$. Thus, after some computations, we get that $m = \frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-\sum_{h\in\mathcal{H}}(B_h/s)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-((\beta-1)|\mathcal{O}|+1)\sum_{h\in\mathcal{H}}(B_h/s)}$. ■

Theorem 3 combined with the results in Table I, provides two approximation algorithms for the JCR problem:

Corollary 2: There exists a *polynomial time* $((11+10c')\frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-\sum_{h\in\mathcal{H}}(B_h/s)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-(|\mathcal{O}|+1)\sum_{h\in\mathcal{H}}(B_h/s)})$ -approximation algorithm and a *quasi-polynomial time* $((\log(n)+(\log(n)-1)c')\frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-\sum_{h\in\mathcal{H}}(B_h/s)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-(\epsilon|\mathcal{O}|+1)\sum_{h\in\mathcal{H}}(B_h/s)})$ -approximation algorithm, $\forall\epsilon > 0$, for the JCR problem, where $n = |\mathcal{A}| + |\mathcal{B}|$.

For example, as ϵ goes to zero, the approximation ratio of Algorithm 2 becomes $\log(n)+(\log(n)-1)c' \approx 7$, in the realistic system setting that $|\mathcal{O}| = 100$, $|\mathcal{H}| = 32$, $B_h = 5, \forall h$, $s = 1$ and $\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki}) = 10,000$. For smaller problem instances, this ratio is even tighter.

Let us now focus on a special case of the JCR problem where the helpers have equal transmission capacities, i.e. $B_h = B, \forall h \in \mathcal{H}$. Notice though that still, 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 IV-A, for this special case of the problem, all the capacities of the facilities are equal, i.e. $S_j = \frac{B}{s}, \forall j \in \mathcal{A}$, except for the facility a_{BS}, which has infinite capacity. a_{BS} can be replaced by $\lceil \frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}\lambda_{ki}}{\lfloor B/s \rfloor} \rceil$ facilities each one of capacity $\frac{B}{s}$. Clearly, the aggregate capacity of them suffices to serve all the demand of the clients of the form b_{kij}, $\forall k, i, j$ and the new instance is equivalent to the initial. In the new instance, all the capacities are equal.

Table I provides six approximation algorithms for the JCR problem, in which the capacities of the helpers are equal. Corollary 3 describes the results:

Corollary 3: There exists an $(\alpha + (\alpha-1)c', (\beta-1)|\mathcal{O}|+1)$-approximation algorithm and an $(\alpha+(\alpha-1)c')\frac{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-|\mathcal{H}|(B/s)}{\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}}(\lambda_{ki})-((\beta-1)|\mathcal{O}|+1)|\mathcal{H}|(B/s)}$-approximation algorithm, for each pair $(\alpha,\beta) \in \{(9,4), (O(1),2),$
$(5,2), (\log n, 1+\epsilon), (10.173, \frac{3}{2}), (30.432, \frac{4}{3})\}$, $\forall \epsilon > 0$ for the uniform-capacities JCR problem, where $n = |\mathcal{A}| + |\mathcal{B}|$.

VI. Conclusion

In this chapter we introduced a novel approximation framework for solving the joint caching and routing problem faced by a mobile network operator. This is a problem of increasing importance in order to achieve a smooth and balanced network response in

the stressful situations that the mobile content demand growth brings. Our model differs from previous works, as it explicitly takes into account the hard bandwidth constraints that bottleneck content delivery when massive content requests are submitted, i.e., during peak traffic hours and in highly populated areas. We provided a set of polynomial time algorithms with various bounded approximation ratios for it's solution. Our methodology is based on a reduction to the $UHCMFL$ variant of the facility location problem. We believe that our work reveals a stronger relation among caching problems and facility location problems and paves the road for exploiting the broad literature that is available for the latter.

REFERENCES

[1] Intel Communications, "Rethinking the Small Cell Business Model" *White Paper*, 2011.

[2] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: Wireless Video Content Delivery through Distributed Caching Helpers", *in Proc. of IEEE Infocom*, 2012.

[3] K. Poularakis, L. Tassiulas, "Optimal Cooperative Content Placement Algorithms in Hierarchical Cache Topologies, *in Proc. of IEEE CISS*, 2012

[4] M.R. Korupolu, C.G. Plaxton, R. Rajaraman, "Placement Algorithms for Hierarchical Cooperative Caching", *in Proc. of ACM-SIAM SODA*, 1999.

[5] S. Borst, V. Gupta, and A. Walid, "Distributed Caching Algorithms for Content Distribution Network", *in Proc. of IEEE Infocom*, 2010.

[6] 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", *Computers and OR*, vol. 35, no. 12, 2008.

[7] J. Dai, Z. Hu, B. Li, J. Liu, and B. Li, "Collaborative Hierarchical Caching with Dynamic Request Routing for Massive Content Distribution", *in Proc. of IEEE Infocom*, 2012.

[8] D. B. Shmoys, E. Tardos, and K. Aardal, "Approximation Algorithms for Facility Location Problems", *in Proc. of ACM STOC*, 1997.

[9] M.R. Korupolu, C.G. Plaxton, R. Rajaraman, "Analysis of a Local Search Heuristic for Facility Location Problems", *in Proc. of ACM-SIAM SODA*, 1998.

[10] Zhang J., Chen B., AND Y. Ye, "A Multi-Exchange Local Search Algorithm for the Capacitated Facility Location Problem", *Mathematics of OR*, vol. 30, no. 2, 2005.

[11] A. Aggarwal, L. Anand, M. Bansal, N. Garg, N. Gupta, S. Gupta, S. Jain, "A 3-Approximation for Facility Location with Uniform Capacities", *in Proc. of IPCO*, 2010.

[12] M. Bateni, M. Hajiaghayi, "Assignment Problem in Content Distribution Networks: Unsplittable Hard-Capacitated Facility Location", *ACM Trans. on Algorithms*, vol. 8, no. 3, 2012.

[13] B. Behsaz, M. Salavatipour, Z. Svitkina, "New Approximation Algorithms for the Unsplittable Capacitated Facility Location Problem", *in Proc. of SWAT*, 2012.