# On the Joint Content Caching and User Association Problem in Small Cell Networks

M. Karaliopoulos, L. Chatzieleftheriou, G. Darzanos, I. Koutsopoulos
Department of Informatics
Athens University of Economics and Business, Athens, Greece
Email: mkaralio@aueb.gr

***Abstract*—Caching at the edge of the radio network is increasingly viewed as a promising countermeasure to the staggering demand for mobile video content. The persistent orientation of newer generations of mobile communication systems towards lower latency and faster radio access speeds only strengthens the arguments in its favor. When content caching is coordinated with other radio resource management functions, in particular, the benefits for the end users and the network operator are significant. In this paper, we investigate these benefits in cache-enabled small cell networks that jointly control $(i)$ the Small-Cell Base Stations (SBSs) that serve as network access points for the mobile users; and $(ii)$ the content that is stored at the SBS co-located caches. Our main contribution is a fast-converging computationally simple heuristic algorithm that iterates between assigning users to small cells and content to SBS caches, to maximize the overall cache hit ratio. The algorithm solutions compete with the optimal assignments at small problem instances and outperform alternative solutions for larger instances, especially when the content demand exhibits spatial locality. Combining good performance with non-prohibitive complexity, the algorithm could become a valuable tool for small cell network operators seeking to optimize the use of radio network resources.**

## I. Introduction

Mobile network operators routinely respond to the growing demand for mobile data with densifying the radio access network. The deployment of various types of Small-cell Base Stations (SBSs) is an expression of this strategy. On the contrary, more disruptive is the proposal to add caches at the SBSs, as a way to alleviate the resulting load of the radio network backhaul links. The rationale behind this proposal, which appears to be fully aligned with the promises for ultra-low latency communications in 5G and beyond mobile cellular networks, is straightforward: if the content stored at the SBS co-located caches matches the demand of the users each time associated with the cell, this demand will be served locally alleviating the traffic load at the backhaul links.

In general, the network possesses two control mechanisms in pursuit of this matching: *content caching*, determining which content is stored at each cache, and *user associations*, determining which cell each user is associated with. The two mechanisms may be exercised over different time scales. For instance, with online caches, the cache placement may be reiterated upon each and every content item request by a user. Alternatively, the cached content may be determined periodically (*e.g.*, every day or half a day) out of estimates for the local content demand and remain fixed for the respective interval of time. On the other hand, a new user association to a small cell may let its cache intact or trigger changes that budget for her individual content demand distribution.

In general, the more aggressively the network carries out these control functions, the better the performance it can achieve (smaller content access delay on the user side, offloading of mobile backhaul) at the expense of computational resources. More recently, in light of the challenging network capacity and latency targets advertised by 5G cellular networks [1], the wireless community has been investigating scenarios of *joint content caching and user association*, where the stored content across the SBS caches and the associations of users to small cells are determined simultaneously.

**Related work:** Hence, in [2] the problem is formulated as an one-to-many matching game between SBSs and users. Users rank SBSs within range according to the radio signal quality, SBSs rank users according to the volume of content they can serve them from their cache, and an adaptation of Shapley's deferred acceptance algorithm [3] matches users with cells. User content preferences are considered identical and align with a single global Zipf distribution over the content catalogue. On the contrary, the two studies in [4] and [5] work with individual user content demand distributions and decouple the joint problem. They both cluster the users according to their content preferences, although they do not clarify how these clusters are assigned to SBSs, in light of the physical space constraints. Then, the cache placements in each cell adapt to the users' preferences through a reinforcement learning algorithm in [4] and by leveraging the Named Data Networking (NDN) technology in [5]. In [6], the problem formulation is very similar to ours (ref. section III). The authors resort to McCormick envelopes to linearize the objective function and Lagrange partial relaxation to decompose the original problem in three linear subproblems. It is not clear whether and how they produce feasible solutions out of the bounds the relaxation produces, while the complexity of the solution technique is prohibitive except for very small problem instances (*e.g.*, two or three small cells and $\sim$10 users).

**Our contributions:** In our work, we revisit the joint content caching and user association problem (JCAP)[1], as this has

[1]Note the difference with the joint content *caching* and *routing* problem, which has been more extensively studied in literature (*e.g.*, [7] [8]). Thereby, users are assumed to be simultaneously associated with all SBSs within range and content items may be routed to them through different cells.

been implicitly or more explicitly formulated in [2], [4], [6]. Compared to [6], in our problem formulation, the objective function draws on the cache hit ratio rather than the content access delay (see section II). Then:

- We propose a heuristic algorithm for JCAP. The algorithm iteratively determines which content to store in SBS caches and which users to associate with small cells, by alternately solving instances of the 0-1 Knapsack and the Generalized Assignment problems, respectively.
- We assess the performance characteristics of this iterative algorithm through a two-stage process. We first compare the algorithm solutions against the optimal ones for toy problem instances that can be solved optimally. Then, we turn to larger problem instances and compare it with two alternative heuristic algorithms under variable user population, cache storage space and cell capacity. The comparison is instructive about the sensitivity of the algorithm to various system parameters.

In section II we present the system model we consider and in section III we formalize the joint content caching and user association problem. We develop our main contributions in sections IV and V and conclude the paper in section VI.

## II. System model

Our system model considers a set of SBSs and a Macro Cell Base station (MBS) that together form a two-layer radio access network. Each SBS can serve the set of mobile users who lie within its range, whereas the MBS can serve the users being within range of all SBSs. We assume that all the SBSs are equipped with caches that can store content items originally accessible through Content Provider platforms and provide them to the users upon request.

***Content Items:*** Let $\mathcal{I}$ be the catalogue of content items that users demand. Content items may have different sizes in bytes. We work with normalized sizes so that $l_i$ denotes the size of item $i$ in multiples of the smallest item and $l_{max}$ is the ratio of the largest-to-smallest item. Replicas of each content item $i \in \mathcal{I}$ may be stored in one or more SBS caches depending on the actual caching decisions. One or more back end server caches hold replicas of the full content catalogue and can serve all those requests that cannot be served by the small cell caches.

***Cells, caches, users:*** Each SBS co-located cache $c \in \mathcal{C}$ has finite storage, $L_c$, measured in multiples of the smallest content item. At any point in time, each cache $c$ stores a finite set of files, referred to as the cache placement $\mathcal{P}_c$.

Each user $u \in \mathcal{U}$ may be located within the range of a different subset of SBSs. We define as $\mathcal{N}(u)$ the "neighborhood" of user $u$, *i.e.*, the set of cells within communication range of user $u$. At any point in time, each user can be associated to only one SBS or the MBS. Users are characterized by individual content preference distributions that express their preferences over the catalogue content items. This distribution can be extracted by relating the thematic categories users are interested in with the thematic characterization of each item (*e.g.*, tags). In our work, we assume that the system is aware of this distribution, *i.e.*, each user $u$ is described by a content preference distribution, $\{p_{ui},\ i \in \mathcal{I}\}$, with $\sum\limits_{i\in\mathcal{I}} p_{ui} = 1$. This distribution is assumed to change much slower than the time scale of user association events, so that it can be considered fixed over multiple user association time epochs. Since the content preferences change both spatially and temporally, the network would need to maintain space- and time-dependent characterizations (profiles) of user content preferences.

The number of users who can associate with an SBS is bounded since the cell resources (bandwidth, transmission power) are finite and associated users consume different parts of these resources. For instance, a user at higher physical distance from the SBS will typically demand higher transmission power on the SBS side for given minimum acceptable transmission rate. Hence, we assume that an SBS incurs a user-to-SBS specific *association cost*, $b_{uc}$, any time user $u \in \mathcal{U}$ associates with SBS $c \in \mathcal{C}$ at minimum acceptable service rate.

In general, the aggregate user association cost for a small cell $c$ is a set function $f : U_c \longrightarrow R$ over the set of users $U_c$ associated to the cell $f$ and may be sub- or super-modular. In this work, we assume that the user association cost is fixed and $f$ is additive, *i.e.*, $f(U_c \cup u_0) = f(U_c) + f(u_0) = \sum\limits_{u\in U_c} b_{uc} + b_{u_0c}$. The number of users that an SBS can serve is limited by the cell capacity $B_c$. Associating a user to an SBS instead of the MBS benefits both the user, who gains access to the locally cached content, and the network, which saves macrocell resources. Hence, a user is associated to the MBS, only when all SBS in the user's "neighborhood" have reached their capacity $B_c$ and cannot serve more users.

## III. The joint content caching and user association problem

### A. Problem formulation

The objective of the joint caching and association problem (JCAP) is to maximize the total demand that can be satisfied across all SBS caches. Let $\{x_{ic}\}$ and $\{y_{uc}\}$ be sets of binary variables, with $x_{ic} = 1$ if item $i$ is stored at the cache of SBS $c$, $y_{uc} = 1$ if user $u$ is associated with SBS $c$, and $x_{ic} = 0$ and $y_{uc} = 0$ in the opposite events, respectively. Then, the demand that can be satisfied across all SBS caches can be written

$$D_{hit} = \sum_{u\in\mathcal{U}} \sum_{c\in\mathcal{N}_u} \sum_{i\in\mathcal{I}} p_{ui} x_{ic} y_{uc} \tag{1}$$

The small cell network operator then, faces the following maximization problem (P1), which we hereafter call the Joint content Caching and user Association (JCA) problem:

$$\begin{aligned}
\max_{\mathbf{x},\mathbf{y}} \quad & D_{hit} & & (P1) \\
s.t. \quad & \sum_{i\in\mathcal{I}} x_{ic} l_i \le L_c, \quad \forall c \in \mathcal{C} & & (2) \\
& \sum_{u\in\mathcal{U}} b_{uc} y_{uc} \le B_c, \quad \forall c \in \mathcal{C} & & (3) \\
& \sum_{c\in\mathcal{N}(u)} y_{uc} \le 1, \quad \forall u \in \mathcal{U}, & & (4) \\
& y_{uc}, x_{ic} \in \{0,1\}, u \in \mathcal{U}, i \in \mathcal{I}, c \in \mathcal{C} & & (5)
\end{aligned}$$

Constraints (2) and (3) reflect the cache storage and cell capacity constraints for each small cell, respectively. Constraint (4) captures the fact that users cannot associate with more than a single SBS within their range. When $y_{uc} = 0$, user $u$ ends up associated with the MBS and fetches all requested content from the Content Provider caches.

The problem (P1) is an instance of bilinear programming, a special class of non-convex quadratic programming. Showing that it is NP-hard is trivial. It suffices to remark that when we fix variables $\{x_{ic}\}$, we get the generalized-assignment type problem first treated in [9]. This problem is equivalent to the maximum Generalized Assignment Problem (GAP)[2], where SBSs correspond to agents (knapsacks) and users to jobs (items) with agent-specific requirements $\{b_{uc}\}$ and profits $\{\sum_{i\in\mathcal{I}} x_{ic}p_{ui}\},\ c \in \mathcal{N}_u$. Since the maximum GAP is an NP-hard problem, the generalization (P1) of its equivalent problem in [9] is NP-hard as well.

## *B. Problem linearization*

The objective function of (P1) is a weighted sum of bilinear terms, *i.e.*, products of binary variables. As a first step towards solving it, we can linearize these products. For each pair of variables $(x_{ic}, y_{uc})$, $i \in \mathcal{I}$, $u \in \mathcal{U}$, $c \in \mathcal{N}_u$, we introduce a new binary variable $z_{iuc} = x_{ic} \cdot y_{uc}$ subject to the additional constraints:

$$z_{uic} \leq x_{ic},\ z_{uic} \leq y_{uc} \text{ and } z_{uic} \geq x_{ic} + y_{uc} - 1 \qquad (6)$$

Plugging $\{z_{iuc}\}$ in (2) and adding constraints (6) to the (P1) formulation, we get a zero-one Linear Program (LP) with $\mathcal{O}(C \cdot I \cdot U)$ additional decision variables and $\mathcal{O}(3C \cdot I \cdot U)$ additional constraints with respect to (P1). This problem, hereafter called (P2), can be tackled with generic Integer LP solvers for small $(C, U, I)$ values. We use these solutions as comparison references for our heuristic in section V-B.

## IV. A HEURISTIC SOLUTION TO JCAP

In this section, we present and analyze a heuristic algorithm for the JCA problem. The algorithm is iterative and essentially decouples the two decisions made in the JCAP, *i.e.*, which SBS should we associate each user with and which content to cache at each cache-enabled SBS. This decoupling is possible at first place since the constraints of (P1) are separable.

## *A. The algorithm*

The algorithm first takes an initialization step, determining the cache content at each SBS when all users within its range are assumed to be associated with it. This assumption implies solving (P1) after removing the user association related constraints (3) and (4) and setting $y_{uc} = 1$ for each SBS $c \in \mathcal{N}_u$ so that the value $f_{ic}$ carried by an item $i$ when placed at cache $c$ is given by

$$f_{ic} = \sum_{u:y_{uc}=1} p_{ui} = \sum_{u} p_{ui}y_{uc} \qquad (7)$$

[2]The inequalities in (4) are replaced by equalities in the maximum GAP. The equivalence of the two problems is shown in *e.g.*, [10], pp. 190-191.

We can then determine the cache placements at the SBS caches by solving independent instances of the 0-1 Knapsack Problem (KP). Namely, at each SBS cache $c$, we solve for

$$\begin{aligned} \max_{\mathbf{x}} \quad & \sum_{i\in\mathcal{I}} f_{ic}x_{ic} && (P3a) \\ s.t. \quad & \sum_{i\in\mathcal{I}} x_{ic}l_i \leq L_c && (8) \\ & x_{ic} \in \{0,1\}, u \in \mathcal{U}, i \in \mathcal{I} && (9) \end{aligned}$$

Then, the algorithm enters an iterative phase. Given the cache placements determined by (P3a), the algorithm updates the user associations at each SBS solving an instance of GAP. Intuitively, users correspond to jobs, SBSs to agents, and each user ("job") bears SBS("agent")-specific costs $b_{uc}$ and profits $f_{uc}$ given by

$$f_{uc} = \sum_{i:x_{ic}=1} p_{ui} = \sum_{i} p_{ui}x_{ic} \qquad (10)$$

where the $\{x_{ic}\}$ values coincide with the solutions of the $C$ (P3a) instances. Hence, at this second step of the heuristic algorithm we solve a single instance of the problem (P3b).

$$\begin{aligned} \max_{\mathbf{y}} \quad & \sum_{u\in\mathcal{U}} \sum_{c\in\mathcal{N}_u} f_{uc}y_{uc} && (P3b) \\ s.t. \quad & \sum_{u\in\mathcal{U}} b_{uc}y_{uc} \leq B_c, \quad \forall c \in \mathcal{C} && (11) \\ & \sum_{c\in\mathcal{N}(u)} y_{uc} \leq 1, \quad \forall u \in \mathcal{U}, && (12) \\ & y_{uc} \in \{0,1\}, u \in \mathcal{U}, c \in \mathcal{C} && (13) \end{aligned}$$

The problem (P3b) iterates on the user associations to SBSs and provides the first feasible solution of the original problem (P1). In turn, the next step of the algorithm revisits the cache placements at each SBS. With the new values of user association variables $\{y_{uc}\}$ computed by (P3b), the algorithm recomputes the item profit values $\{f_{ic}\}$ in (7) and solves anew the $C$ 0-1 KP instances (P3a).

These two steps of the algorithm, iterating separately on cached content and the user associations through problems (P3a) and (P3b), respectively, are repeated until no improvement is achievable in terms of the aggregate demand $D_{hit}$ in (1) that can be satisfied by the cache content.

## *B. Algorithm correctness and computational complexity*

In the initialization step, the algorithm generates a non-feasible solution for (P1), violating constraint (4) unless $|\mathcal{N}_u = 1|, \forall u \in \mathcal{U}$, and possibly constraint (3). The objective function value for this solution provides an upper bound for the optimal solution, $OPT_{JCAP}$.

The algorithm generates feasible solutions for JCAP as soon as it enters the iterative phase. The value of the objective function upon the first execution of (P3b) is smaller than both the bound computed upon the initialization step and $OPT_{JCAP}$. Subsequent iterations get new feasible solutions solving, alternately, either the $C$ 0-1 KPs (P3a) or the GAP

(P3b). The solution produced in each iteration is checked against the current solution, which is the best feasible solution achieved so far, and replaces it as far as it improves over it. Since the optimal value is finite, the algorithm will definitely terminate with a solution that is upper bounded by $OPT_{JCAP}$.

From a computational point of view, the heuristic algorithm proceeds through two types of iterations, on the user associations and cached content, respectively. Each iteration on the cached content demands the solution of $C$ 0-1 KP instances. Solving them with the pseudo-polynomial Dynamic Programming (DP) algorithm (see, for example, [11], 6.4) the time complexity is $O(CIL_c)$. Each iteration on the user associations demands the solution of one GAP. With the 2-approximation algorithm in [12][3] the required time is $O(CIL_c + CI)$. Hence, the time complexity of the iterative heuristic algorithm is $O(kCIL_c)$, where $k$ is the number of iterations the algorithm demands. Indicatively, in all experiments reported in section V, $k < 10$.

## V. Evaluation of the heuristic

### A. Methodology

The evaluation of the heuristic proceeds in two steps. First, for small problem instances, we compare the heuristic against the optimal solution, as this results from an exact generic ILP solver. In the absence of analytical results about the approximability properties of the heuristic, this evaluation step provides important evidence about the quality of the solution it produces. Then, we shift to more realistic problem instances and compare the heuristic against simpler alternatives for tackling the user association and content caching tasks. This step quantifies the achievable performance gain with our heuristic.

In all experiments, we let the item sizes vary randomly in $\{1, l_{max}\}$ and the association costs in $\{1, b_{max}\}$, where $l_{max}$ and $b_{max}$ are the maximum over minimum ratios of the two quantities, respectively. With respect to the content demand distributions of individual users, we consider two scenarios. In the first one, the probabilities $\{p_{ui}\}$ are randomly assigned to end users. In the second one, end users are partitioned according to their physical location, so that users within range of approximately the same BSs belong to the same subset. Users within each of these $N_{cl}$ subsets are assigned identical demand distributions over the content items, *i.e.*, there are $N_{cl}$ distinct content demand distributions, one per user subset. Although the two scenarios for the user content demand distributions serve as rather extreme cases regarding the *spatial locality* of content demand, they are most informative about the properties of our heuristic and the way it compares with alternative solutions, as discussed in section V-C.

### B. Small-scale problem instances: optimality

In the first set of experiments we consider two cells ($C = 2$), fix the number of items to $I = 100$ and let the number of users vary in $\{4, 9\}$. The size of the normalized cache capacity $L_n = \frac{L_c}{I \cdot L_{max}/2}$ ranges in [0.1, 0.5], where $I \cdot L_{max}/2$ is the expected normalized catalog size. Figures 1a and 1b compare the cache hit rate, $HR_{heur}$, achieved by our iterative heuristic against the optimal one, $HR_{opt}$, as produced by a generic ILP solver. Each point in the plot averages 20 different instances of the experiment, *i.e.*, sets of item sizes $\{L_i\}$ and association costs $\{b_{uc}\}$. In all cases, the heuristic solution lies very close to the optimal one. To quantify this, let $\Delta H = HR_{opt} - HR_{heur}$ be the empirical gap between the two solutions and $G = \frac{\Delta H}{HR_{opt}}$x$100\%$ denote its $\%$ normalized value. The median of $G$ over all experiments is practically $0\%$ under both types of content demand, the two solutions coinciding in more than half of the experiments. The worst-case deviation from the optimal in all experiments is less than 30%, as shown in Table I.

TABLE I
Accuracy of the iterative heuristic.

| Comparison scenario | var Users rand. demand | var Users clust. demand | var Items rand. demand |
|---|---|---|---|
| median $\Delta H(G)$ | 0.001(0.2%) | 0(0%) | 0.004(0.81%) |
| 95$^{th}$ perc. $\Delta H(G)$ | 0.097(17.7%) | 0.1(14.2%) | 0.081(13.3%) |
| max $\Delta H(G)$ | 0.161(29.2%) | 0.21(25%) | 0.189(35.2%) |

In the second set of experiments, we consider $C = 3$ cells, fix the number of users to $U = 8$ and let the number of content items vary in $[40, 80]$ in 10-item step sizes. The capacities of the three cells are set to $B_c = 20$ and their cache storage space varies so that $L_n \in [0.1, 0.6]$. Again, as shown in Fig. 1c, the heuristic solution approximates well the optimal cache hit ratio. The median of the deviation is approximately zero and the worst-case (max) value of G is $35\%$.

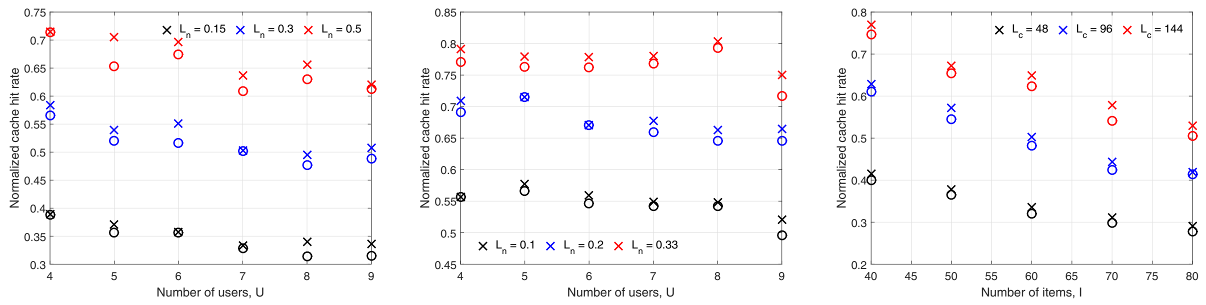
### C. Performance gain and sensitivity analysis

We now turn into more realistic system scenarios and compare our heuristic with two alternative solutions for JCAP.

*Greedy heuristic:* The first one is a greedy-like heuristic. The algorithm parses all user content demand probabilities in order of decreasing size. Each parsed probability corresponds to given user $u_0$ and item $i_0$. If the user is not yet associated to a cell, the algorithm seeks to associate the user with the cell of minimum association cost out of those that fulfill the cache capacity constraints and user association constraints, as given by Eqs. (2) and (3), respectively. At the same time, it caches item $i_0$ to the corresponding cache. If the user is already associated with a cell, the algorithm checks whether item $i_0$ is already in the cache of that cell; if not, it is added to it as far as the cache capacity constraint (2) is satisfied.

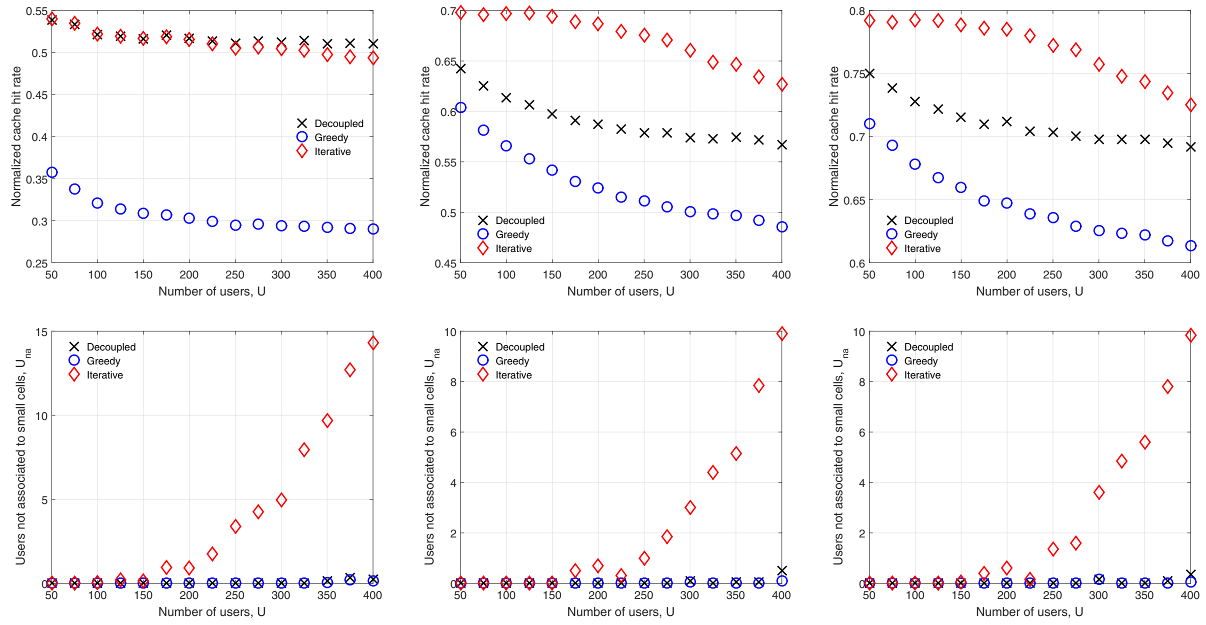
*Decoupled user associations:* With this heuristic, abbreviated hereafter as *Decoupled*, the user association decisions are separated from the content caching ones and timewise precede them. First, the cell association options for each user are ranked in order of increasing association cost (*cell preference list*). Then, users are listed in order of increasing minimum association cost. The heuristic parses users sequentially and tries to associate them with the cell in their cell preference list that presents the minimum association cost. If this not possible for a user $u$, *i.e.*, the user association constraint in (4)

[3]The algorithm decomposes the GAP into a series of 0-1 KPs. These approximation and time complexity scores assume use of the pseudo-polynomial DP algorithm for solving those KPs.



a. I=100, C=2, random demand b. I=100, C=2, spatially clustered demand $N_{cl} = 10$ c. U=8, C=3, random demand

Fig. 1. Comparison of the iterative heuristic ('o') with the optimal solution ('x'). Marks correspond to averages of 15 simulation runs. $b_{max} = 10$, $B_c = 25$.



a. No spatial locality, $L_n = 0.3$ b. Spatial locality $N_{cl} = 10$, $L_n = 0.1$ c. Spatial locality $N_{cl} = 10$, $L_n = 0.2$

Fig. 2. Comparison of the three heuristics as a function of the number of users: $I$=1000, $C$=20, $B_c$=200, $l_{max}$=12, $b_{max}$=20.

.

is violated if $u$ associates, the next cell in $u$'s cell preference list is checked and so on till a user is associated with a cell or its cell preference list is exhausted.

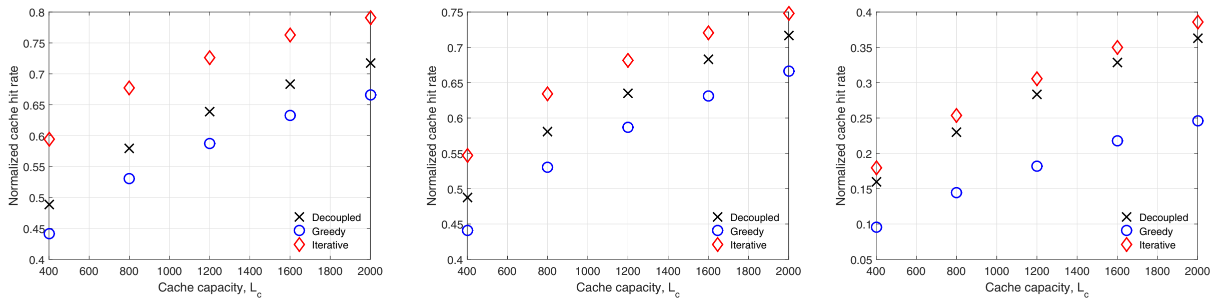
We compare the performance of the three heuristics and how this depends on a number of system parameters.

**Impact of spatial locality in content demand:** The first comparison focuses on the two alternatives for generating the user content demand distributions, described in section V-A. In this set of experiments, the SBS cache sizes correspond to roughly 15% of the expected content catalogue size ($I \cdot l_{max}/2$) and the expected number of users ($2B_c/b_{max}$) that can fit in one small cell is 20. Under no spatial locality in demand, the decoupled heuristic competes with our iterative heuristic, and even slightly outperforms it for high numbers of users, as shown in Fig. 2a. At those user levels, the decoupled approach can match all users to SBSs, whereas our iterative heuristic is forced to associate a few with the MBS. The non-satisfied demand of these users corresponds to the marginal performance gain of the decoupled heuristic. The greedy heuristics cannot compete with either of the two alternatives.

When there is spatial locality, the performance of all three heuristics improves, the greedy one benefiting most (Fig. 2b). In this scenario, it matters more *which* users will be grouped in a cell rather than *how many*. Factoring the content preferences of users in its decisions, the iterative heuristic clearly outperforms the decoupled one. This performance gain fades out as the number of users grows since the latter manages to squeeze more users in small cells.

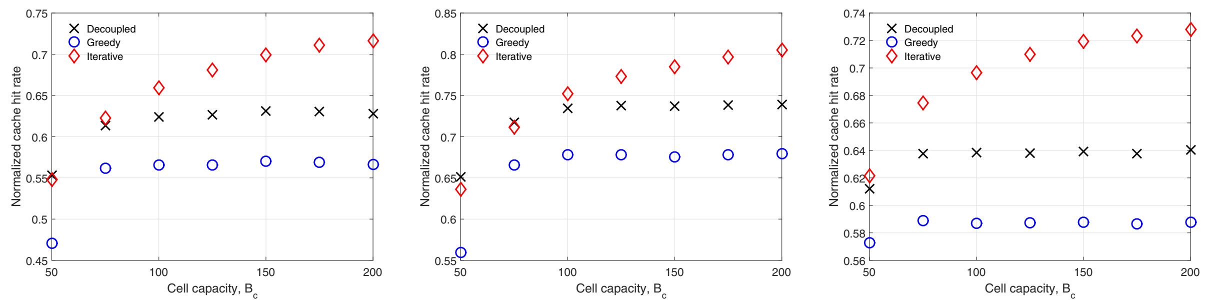
**Impact of cache sizes:** In this second set of experiments, we only consider content demand with embedded spatial locality. The capacity of each cell corresponds to an expected number of 20 users so that the 20 (30) small cells in Fig. 3a(b) essentially render the user association constraints inactive for the considered 200 users. On the contrary, we activate the cache capacity constraint by letting the cache size vary from 5% to 15% of the overall content catalogue size.

In the absence of user association constraints, the iterative heuristic clearly outperforms the other two. Its gain scales up both within and *across* the three scenarios in Fig. 3, as the network resources (cell capacity, cache size) grow.



a. $B_c$ = 100, $l_{max}$=12.  b. $B_c$ = 250, $l_{max}$=12.  c. $B_c$ = 100, $l_{max}$=20.

Fig. 3. Comparison of the three heuristics as a function of the cache size: $I$=2000, $C$=30, $U$=200, $N_{cl}$=10, $b_{max}$=25.



a. $U$=300, $L_n$=0.1  b. $U$=300, $L_n$=0.3  c. $U$=200, $L_n$=0.1

Fig. 4. Comparison of the three heuristics as a function of the cell capacity: $C$=30, $N_{cl}$=10, $l_{max}$=12, $b_{max}$=20.

**Impact of cell capacities:** In the third set of experiments, the active constraints relate to user associations. Cache sizes are generously set to $15\%$ of the content catalogue, whereas the cell capacities vary from the expected equivalent of 5 up to 20 users. Fig. 4 reinforces what we have touched upon in Fig. 2b. Prioritizing the associations of users to the minimum-cost cells and adapting *a posteriori* the cached content to the user population, the decoupled heuristic ends up serving considerably more users and content demand through small cells. As more capacity is added to small cells and the user association constraints are relaxed, our iterative heuristic can associate more users with similar demand to the same cells, thus serving more demand locally.

## VI. Conclusions

Our work adds to the research thread on the joint content caching and user association problem in small cell networks. The problem has attracted interest over the last few years, in light of recent trends in the area of edge caching and 5G networks. In particular, we have proposed a fast converging iterative heuristic algorithm that combines good performance with non-prohibitive complexity. We have compared its performance to plausible alternative heuristics and demonstrated its sensitivity to the sparseness of cell capacity and storage resources, crafting arguments for its practical relevance.

## Acknowledgement

This research has been funded by the Operational Program ”Human Resources Development, Education and Lifelong Learning”, co-financed by European Union (EU) and Greek national funds. I. Koutsopoulos acknowledges support from a GSRT Research Reinforcement grant for the EU R&D project netCommons, and from the AUEB grant “Original Scientific Publications 2019”.

## References

[1] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, “What Will 5G Be?” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, June 2014.

[2] F. Pantisano, M. Bennis, W. Saad, and M. Debbah, “Cache-aware user association in backhaul-constrained small cell networks,” in *Proc. WiOpt*, Hammamet, Tunisia, May 2014, pp. 37–42.

[3] D. Gale and L. S. Shapley, “College admissions and the stability of marriage,” *Amer. Math. Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[4] M. ElBamby, M. Bennis, W. Saad, and M. Latva-Aho, “Content-aware user clustering and caching in wireless small cell networks,” in *Proc. Int’l Symp. on Wireless Commun. Sys. (ISWCS)*, 2014, pp. 945–949.

[5] A. V. Ribeiro, L. N. Sampaio, and A. Ziviani, “Affinity-based user clustering for efficient edge caching in content-centric cellular networks,” in *in Proc. ISCC*, Lisbon, Portugal, 2018.

[6] Y. Wang, X. Tao, X. Zhang, and G. Mao, “Joint caching placement and user association for minimizing user download delay,” *IEEE Access*, vol. 4, pp. 8625–8633, December 2016.

[7] M. Dehghan, B. Jiang, A. Seetharam, T. He, T. Salonidis, J. Kurose, D. Towsley, and R. Sitaraman, “On the Complexity of Optimal Request Routing and Content Caching in Heterogeneous Cache Networks,” *IEEE/ACM Trans. on Netw.*, vol. 25, no. 3, pp. 1635–1648, Jun. 2017.

[8] A. Khreishah, J. Chakareski, and A. Gharaibeh, “Joint caching, routing, and channel assignment for collaborative small-cell cellular networks,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 8, pp. 2275–2284, August 2016.

[9] L. Chalmet and L. Gelders, “Lagrangian relaxations for a generalized assignment-type problem,” in *Proc. Second European Congress on Operations Research*, Stockholm, Sweden, Nov-Dec 1976, pp. 103–109.

[10] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York, NY, USA: John Wiley & Sons, Inc., 1990.

[11] S. Dasgupta, C. H. Papadimitriou, and U. Vazirani, *Algorithms*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 2008.

[12] R. Cohen, L. Katzir, and D. Raz, “An efficient approximation for the generalized assignment problem,” *Information Processing Letters*, vol. 100, no. 4, pp. 162 – 166, 2006.