

A Low-Complexity Approach to Distributed Cooperative Caching with Geographic Constraints

Konstantin Avrachenkov, Jasper Goseling, Berksan Serbetci

▶ **To cite this version:**

Konstantin Avrachenkov, Jasper Goseling, Berksan Serbetci. A Low-Complexity Approach to Distributed Cooperative Caching with Geographic Constraints. Proceedings of the ACM on Measurement and Analysis of Computing Systems , 2017, 1, pp.27:1 - 27:24. 10.1145/3084465 . hal-01648129

HAL Id: hal-01648129
https://inria.hal.science/hal-01648129

Submitted on 24 Nov 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Low-Complexity Approach to Distributed Cooperative Caching with Geographic Constraints

KONSTANTIN AVRACHENKOV, INRIA Sophia Antipolis
JASPER GOSELING, University of Twente
BERKSAN SERBETCI, University of Twente

CCS Concepts: • **Networks** → **Wireless access points, base stations and infrastructure**; *Network performance evaluation*; *Network dynamics*; *Mobile networks*; Network resources allocation; • **Theory of computation** → *Mathematical optimization*; *Algorithmic game theory and mechanism design*;

Additional Key Words and Phrases: Caching; Wireless networks; Distributed optimization; Game theory; Simulated annealing

ACM Reference format:
Konstantin Avrachenkov, Jasper Goseling, and Berksan Serbetci. 2017. A Low-Complexity Approach to Distributed Cooperative Caching with Geographic Constraints. *Proc. ACM Meas. Anal. Comput. Syst.* 1, 1, Article 27 (June 2017), 24 pages.
DOI: http://dx.doi.org/10.1145/3084465

1 INTRODUCTION

Data traffic in cellular networks is rapidly expanding and is expected to increase so much in the upcoming years that existing network infrastructures will not be able to support this demand. One of the bottlenecks will be formed by the backhaul links that connect base stations to the core network and, therefore, we need to utilize these links as efficiently as possible. A promising means to increase efficiency compared to existing architectures is to proactively cache data in the base stations. The idea is to store part of the data at the wireless edge and use the backhaul only to refresh the stored data. Data replacement will depend on the users' demand distribution over time. As this distribution is varying slowly, the stored data can be refreshed at off-peak times. In this way, caches containing popular content serve as helpers to the overall system and decrease the maximum backhaul load.

Our goal in this paper is on developing low-complexity distributed and asynchronous content placement algorithms. This is of practical relevance in cellular networks in which an operator wants to optimize the stored content in caches (i.e., base stations) while keeping the communication in the network to a minimum. In that case it will help that caches exchange information only locally.

In the remainder of the introduction we shall give an overview of the model and contributions. Then, in the ensuing section, we provide a very thorough discussion of the related works.

We consider continuous and discrete models with caches located at arbitrary locations either in the plane or in the grid. Caches know their own coverage area as well as the coverage areas of other caches that overlap with this region. There is a content catalog from which users request files according to a known probability distribution. Each cache can store a limited number of files and the goal is to minimize the probability that a user at an arbitrary location in the plane will not find the file that she requires in one of the caches that she is covered by. We develop low-complexity asynchronous distributed cooperative content placement caching algorithms that require communication only between caches with overlapping coverage areas. In the basic algorithm, at each iteration a cache will selfishly update its cache content by minimizing the local miss probability and by considering the

© 2017 ACM. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *Proc. ACM Meas. Anal. Comput. Syst.*, https://doi.org/http://dx.doi.org/10.1145/3084465.

content stored by neighbouring caches. We provide a game theoretic perspective on our algorithm and relate the algorithm to the best response dynamics in a potential game. We demonstrate that our algorithm has polynomial step update complexity (in network and catalog size) and has overall convergence in polynomial time. This does not happen in general in potential games. We also provide two simulated annealing-type algorithms (stochastic and deterministic) that find the best equilibrium corresponding to the global minimum of the miss probability. Finally, we illustrate our results by a number of numerical results with synthetic and real world network models.

To specify, our contributions are as follows:

- We provide a distributed asynchronous algorithm for optimizing the content placement which can be interpreted as giving the best response dynamics in a potential game;
- We prove that the best response dynamics can be obtained as a solution of a convex optimization problem;
- We prove that our algorithm converges and establish polynomial bounds (in terms of network as well as catalog size) on the running time and the complexity per iteration;
- We evaluate our algorithm through numerical examples using a homogeneous spatial Poisson process and base station locations from a real wireless network for the cellular network topology. We study the miss probability evolution on real and synthetic networks numerically and show that our distributed caching algorithm performs significantly better than storing the most popular content or probabilistic content placement policies or adhoc multi-LRU cooperative policy. We observe that as the coordination between caches increases, our distributed caching algorithm's performance significantly improves;
- In fact, we demonstrate that in most cases of practical interest the algorithm based on best response converges to the globally optimal content placement;
- Finally, we present simulated annealing type extensions of our algorithm that converge to the globally optimal solution. Our simulated annealing algorithms have efficient practical implementations that we illustrate by numerical examples.

Let us outline the organization of the paper. In Section 2, we provide a thorough review of the relevant works. In Section 3 we give the formal model and problem definitions. In Section 4 we provide the game formulation of the problem, analyze the structures of the best response dynamics and Nash equilibria, provide bounds on the rate of convergence and analyze the computational complexity of the content placement game. In Section 5 we give a content placement game example converging to local optimum and provide some remedial algorithms, such as stochastic and deterministic simulated annealing, to achieve global optimum. In Section 6, we present practical implementations of our low-complexity algorithms and show the resulting performances for various network topologies. In Section 7 we conclude the paper with some discussions and provide an outlook on future research.

2 RELATED WORK

Caching has received a lot of attention in the literature. Here we provide an overview of the work that is most closely related to the current paper. Namely, we survey the works about systems (networks) of caches.

Building upon the approximation technique from Dan and Towsley [13], in [40] Rosensweig et al. proposed an approximation technique for a network of caches with general topology. Unfortunately, it is not easy to provide the performance guarantees of that approximation. Using the characteristic time approximation [16] (see also [21, 22]), Che et al. [12] provide a very accurate approximation for the cache networks with tree hierarchy. The characteristic time approximation is intimately related to the TTL-cache systems [18–20]. Then, in [23] Garetto et al. have shown how one can extend and refine the characteristic time approximation technique to describe quite general networks of caches. The recent upsurge in interest in the analysis of cache networks is motivated by two important application domains: Content Delivery Networks (CDN) [6, 9, 30] and Information Centric Networks (ICN) [11, 41, 48].

By now there is a significant body of literature on caching in wireless networks. A general outline of a distributed caching architecture for wireless networks has been presented in a series of works [25, 26, 43]. Specifically, Shanmugam et al. [43] consider a model in which a bipartite graph indicates how users are connected to base stations. It is shown that minimizing delay by optimally placing files in the base stations is an NP-complete problem and a factor $1/2$ approximation algorithm is developed. Furthermore, it is shown that a coded placement can be obtained through linear programming. Poularakis et al. [38] provide an approximation algorithm for the uncoded placement problem by establishing a connection to facility location problems.

Going beyond modelling the geometry of the problem by a bipartite graph, [1], [8], and [4] consider placement of base stations in the plane according to a stochastic geometry. These works consider a probabilistic content placement strategy in which each base station independently of the other base stations stores a random subset of the files. In [1] coded and uncoded placement are compared for various performance measures. In [4] dynamic programming approach is developed to find the optimal coded placement. The authors of [8] show how the placement with the average storage capacity constraint can be mapped to the placement with the hard storage constraint.

Similar to the current work, [10] considers optimal uncoded content placement when caches are located at arbitrary positions. It is argued that this is an NP-complete problem and a Gibbs sampling based approach is developed that converges to the globally optimal placement strategy. However, there are important differences between [10] and the present work. In the present work we deal directly with the miss probability minimization and cast the problem into the framework of potential games. This allows us to obtain an algorithm that converges in polynomial time to a Nash equilibrium. Of course, we cannot guarantee that our basic algorithm converges to the best Nash equilibrium but in most practical scenarii it does so. We also show that when our algorithm converges to a local optimum the resulting performance gap in comparison with the global optimum is very small. Finally, to find global optimum, we provide generalized algorithms based on stochastic [28] and deterministic [39] annealing.

In [44] game theory is used to establish incentives for users in the network to locally perform caching. Bastug et al. [7] couple the caching problem with the physical layer, considering the SINR and the target bit rate. In [32] it is demonstrated that caching in base station can increase the efficiency in the wireless downlink by enabling cooperation between base stations. In [42] and [27] caching at the base stations in a FOG-RAN architecture is considered and bounds on the minimum delivery latency are established. Dehghan et al. [14] propose utility-driven caching and develop online algorithms that can be used by service providers to implement various caching policies based on arbitrary utility functions. Neglia et al. [35] show that even linear utilities help to cover quite a number of interesting particular cases of cache optimization. It is interesting to note that the authors of [35] have also used stochastic simulated annealing type algorithms for the solution of the cache utility optimization problem. Ioannidis et al. [29] propose a mechanism for determining the caching policy of each mobile user that maximizes the system's social welfare in the absence of a central authority. Mohari et al. [33], demonstrate the benefits of learning popularity distribution and it is a good direction for the extension of the present approach.

One more view on caching in networks is given by Maddah-Ali and Niesen [31] who consider a model in which content is cached by users who are connected to the core network by a shared link and establish information-theoretic bounds on backhaul rate that is required to satisfy requests for files by users. In a related study, it is demonstrated [49] that caching relaxes the constraints on the wireless channel state information that is required.

3 MODEL AND PROBLEM DEFINITION

We consider a network of N base stations that are located in the plane $\mathbb{R}^2$. We will use the notation $[1:N] = \{1,\dots,N\}$ and $\Theta = \mathbb{P}([1:N]) \setminus \emptyset$, where $\mathbb{P}([1:N])$ is the power set of $[1:N]$. Each base station is covering a certain region of the plane and we specify the geometric configuration of the network through A_s, $s \in \Theta$, which

denotes the area of the plane that is covered only by the caches in subset s, namely $A_s = (\cap_{\ell \in s} \bar{A}_\ell) \cap (\cap_{\ell \notin s} \bar{A}_\ell^c)$, where $\bar{A}_\ell$ is the complete coverage region of cache ℓ.

As a special case we will consider the case that all base stations have the same circular coverage region with radius r. In this case we specify the location of each base station, with x_m for the location of base station $m \in [1:N]$. We then obtain $\bar{A}_m$ as the disc of radius r around x_m.

Each base station is equipped with a cache that can be used to store files from a content library $C = \{c_1, c_2, \ldots, c_J\}$, where $J < \infty$. Each element c_j is represents a file. All files in the content library are assumed to have the same size. Caches have capacity K, meaning that they can store K files. For clarity of presentation, we assume homogeneous capacity for the caches. However, our work can immediately be extended to the network topologies where caches have different capacities.

Our interest will be in a user in a random location in the plane, uniformly distributed over the area that is covered by the N base stations, *i.e.*, uniformly distributed in $A_{cov} = \cup_{s \in \Theta} A_s$. The probability of a user in the plane being covered by caches $s \in \Theta$ (and is not covered by additional caches) is denoted by $p_s = |A_s|/|A_{cov}|$. A user located in A_s, $s \in \Theta$ can connect to all caches in subset s.

The user requests one of the files from the content library. The aim is to place content in the caches ahead of time in order to maximize the probability that the user will find the requested file in one of the caches that it is covered by.

The probability that file c_j is requested is denoted as a_j. Without loss of generality, $a_1 \geq a_2 \geq \cdots \geq a_J$. Even though any popularity distribution can be used, most of our numerical results will be based on the Zipf distribution. Newman shows that the probability of requesting a specific file from Internet content, *i.e.*, the popularity distribution of a content library, can be approximated by using the Zipf distribution [37] . The probability that a user will ask for content c_j is then equal to

$$a_j = \frac{j^{-\gamma}}{\sum_{j=1}^{J} j^{-\gamma}}, \tag{1}$$

where $\gamma > 0$ is the Zipf parameter.

Content is placed in caches using knowledge of the request statistics $a_1, \ldots, a_J$, but without knowing the actual request made by the user. We denote the placement policy for cache m as

$$b_j^{(m)} := \begin{cases} 1, & \text{if } c_j \text{ is stored in cache } m, \\ 0, & \text{if } c_j \text{ is not stored in cache } m, \end{cases} \tag{2}$$

and the overall placement strategy for cache m as $\mathbf{b}^{(m)} = \left[b_1^{(m)}, \ldots, b_J^{(m)}\right]$ as a J-tuple. The overall placement strategy for the network is denoted by $\mathbf{B} = \left[\mathbf{b}^{(1)}; \ldots; \mathbf{b}^{(N)}\right]$ as an $J \times N$ matrix.

Our performance metric $f(\mathbf{B})$ is the probability that the user does not find the requested file in one of the caches that she is covered by, i.e.,

$$f(\mathbf{B}) = \sum_{j=1}^{J} a_j \sum_{s \in \Theta} p_s \prod_{\ell \in s} (1 - b_j^{(\ell)}). \tag{3}$$

And our goal is to find the optimal placement strategy minimizing the total miss probability as follows:

Problem 1.

$$\begin{aligned} &\min \ f(\mathbf{B}) \\ &\mathbf{s.t.} \quad b_1^{(m)} + \cdots + b_J^{(m)} = K, \quad b_j^{(m)} \in \{0,1\}, \quad \forall j, m. \end{aligned} \tag{4}$$

We will provide a distributed asynchronous algorithm to address Problem 1 in which we iteratively update the placement policy at each cache. We will see that this algorithm can be viewed as the best response dynamics in a

potential game. We will make use of the following notation. Denote by $\mathbf{b}^{(-m)}$ the placement policies of all players except player m. We will write $f(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)})$ to denote $f(\mathbf{B})$. Also, for the sake of simplicity for the potential game formulation that will be presented in the following section, let $f^{(m)}$ denote the miss probability for a user that is located uniformly at random within the coverage region of cache m, *i.e.*,

$$f^{(m)}(\mathbf{B}) = \sum_{j=1}^{J} a_j (1 - b_j^{(m)}) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)})$$

$$= \sum_{j=1}^{J} a_j (1 - b_j^{(m)}) q_m(j), \tag{5}$$

where

$$q_m(j) = \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}). \tag{6}$$

4 POTENTIAL GAME FORMULATION

In this section we provide a distributed asynchronous algorithm to address Problem 1 in which we iteratively update the placement policy at each cache. We will see that this algorithm can be formulated as providing the best response dynamics in a potential game.

The basic idea of our algorithm is that each cache tries selfishly to optimize the payoff function $f^{(m)}(\mathbf{b}^{(m)})$ defined in (5). That is, given a placement $\mathbf{b}^{(-m)}$ by the other caches, cache m solves for $\mathbf{b}^{(m)}$ in

Problem 2.

$$\begin{aligned} &\min \ f^{(m)}\left(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}\right) \\ &\mathbf{s.t.} \quad b_1^{(m)} + \cdots + b_J^{(m)} = K, \quad b_j^{(m)} \in \{0, 1\}, \quad \forall j. \end{aligned} \tag{7}$$

Each cache continues to optimize its placement strategy until no further improvements can be made, that is until no player can take an advantage from the other players. At this point, $\mathbf{B}$ is a *Nash equilibrium* strategy, satisfying

$$f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}) \leq f^{(m)}(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}), \quad \forall m, \tilde{\mathbf{b}}^{(m)}. \tag{8}$$

We will refer to this game as the *content placement game* and demonstrate in the next subsection that the introduced game is a potential game [34] with many nice properties.

4.1 Convergence analysis

In this subsection we prove that if we allow caches to repeatedly update their caches we are guaranteed to converge to a Nash equilibrium in finite time. The order in which caches are scheduled to update their strategy is not important, as long as all caches are scheduled infinitely often.

Theorem 1. *The content placement game defined by payoff functions (5) is a potential game with the potential function given in (3). Furthermore, if we schedule each cache infinitely often, the best response dynamics converges to a Nash equilibrium in finite time.*

Proof. To show that the game is potential with the potential function $f(\mathbf{B})$, we need to check that

$$f^{(m)}(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}) = f(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - f(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}).$$

Now,

$$\begin{aligned} f^{(m)}(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}) &= \sum_{j=1}^{J} a_j (1 - \tilde{b}_j^{(m)}) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}) \\ &\quad - \sum_{j=1}^{J} a_j (1 - b_j^{(m)}) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}) \\ &= \sum_{j=1}^{J} a_j \left(b_j^{(m)} - \tilde{b}_j^{(m)} \right) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}), \end{aligned}$$

and

$$f(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - f(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}) = \sum_{j=1}^{J} a_j \sum_{s \in \Theta} p_s \prod_{\ell \in s} (1 - \tilde{b}_j^{(\ell)}) - \sum_{j=1}^{J} a_j \sum_{s \in \Theta} p_s \prod_{\ell \in s} (1 - b_j^{(\ell)}).$$

Since $f(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - f(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}) = 0$ when $m \notin s$,

$$\begin{aligned} f(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - f(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}) &= \sum_{j=1}^{J} a_j \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s} (1 - \tilde{b}_j^{(\ell)}) - \sum_{j=1}^{J} a_j \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s} (1 - b_j^{(\ell)}) \\ &= \sum_{j=1}^{J} a_j (1 - \tilde{b}_j^{(m)}) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}) \\ &\quad - \sum_{j=1}^{J} a_j (1 - b_j^{(m)}) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}) \\ &= \sum_{j=1}^{J} a_j \left(b_j^{(m)} - \tilde{b}_j^{(m)} \right) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}) \\ &= f^{(m)}(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}), \end{aligned}$$

which completes the proof of the first statement.

Since we only have a finite number of placement strategies, will not miss any cache in the long-run and in a potential game each non-trivial best response provides a positive improvement in the potential function, we are guaranteed to converge to a Nash equilibrium in finite time. □

4.2 Structure of the best response dynamics

In this subsection we will analyze the structure of the best response dynamics. More precisely, we demonstrate that a solution to Problem 2 can be obtained by solving a convex, in fact linear, optimization problem and we provide the solution in closed form.

First we present the relaxed version of Problem 2 as follows. Given a placement $\mathbf{b}^{(-m)}$ by the other caches, cache m solves for $\mathbf{b}^{(m)}$ in

Problem 3.

$$\begin{aligned} &\min \ f^{(m)}\left(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}\right) \\ &\textbf{s.t.} \quad b_1^{(m)} + \cdots + b_J^{(m)} = K, \quad b_j^{(m)} \in [0,1], \quad \forall j. \end{aligned} \tag{9}$$

Note that in Problem 3, $b_j^{(m)}$ can now take values from the interval $[0,1]$ instead of the set $\{0,1\}$ which allows us to present the following lemma.

Lemma 1. *Problem 3 is a convex, in fact linear, optimization problem.*

Proof. It follows immediately from (5). □

We already showed that Problem 3 is convex by Lemma 1 and the constraint set is linear as given in (9). Thus KKT conditions provide necessary and sufficient conditions for optimality. The Lagrangian function corresponding to Problem 3 becomes

$$L\left(\mathbf{b}^{(m)}, \nu, \eta, \omega\right) = \sum_{j=1}^{J} a_j (1 - b_j^{(m)}) q_m(j) + \nu \left(\sum_{j=1}^{J} b_j^{(m)} - K\right) - \sum_{j=1}^{J} \eta_j b_j^{(m)} + \sum_{j=1}^{J} \omega_j \left(b_j^{(m)} - 1\right), \tag{10}$$

where $\mathbf{b}^{(m)}$, η, $\omega \in \mathbb{R}_+^J$ and $\nu \in \mathbb{R}$.

Let $\bar{\mathbf{b}}^{(m)}$, $\bar{\eta}$, $\bar{\omega}$ and $\bar{\nu}$ be primal and dual optimal. The KKT conditions for Problem 3 state that

$$\sum_{j=1}^{J} \bar{b}_j^{(m)} = K, \tag{11}$$

$$0 \le \bar{b}_j^{(m)} \le 1, \quad \forall j = 1, \ldots, J, \tag{12}$$

$$\bar{\eta}_j \ge 0, \quad \forall j = 1, \ldots, J, \tag{13}$$

$$\bar{\omega}_j \ge 0, \quad \forall j = 1, \ldots, J, \tag{14}$$

$$\bar{\eta}_j \bar{b}_j^{(m)} = 0, \quad \forall j = 1, \ldots, J, \tag{15}$$

$$\bar{\omega}_j \left(\bar{b}_j^{(m)} - 1\right) = 0, \quad \forall j = 1, \ldots, J, \tag{16}$$

$$-a_j q_m(j) + \bar{\nu} - \bar{\eta}_j + \bar{\omega}_j = 0, \quad \forall j = 1, \ldots, J. \tag{17}$$

The next result demonstrates that the optimal solution of the relaxed local optimization problem follows a threshold strategy for each cache. As in the global optimization case, files are ordered according to a function of the placement policies of the neighbouring caches and then the first K files are stored. Contrary, to the case of global optimization, this solution is obtained explicitly, because the placement strategies of the other caches are assumed constant.

Theorem 2. *The optimal solution to Problem 3 is given by*

$$\bar{b}_j^{(m)} = \begin{cases} 1, & \text{if } \pi_m^{-1}(j) \le K, \\ 0, & \text{if } \pi_m^{-1}(j) > K, \end{cases} \tag{18}$$

where $\pi_m : [1, J] \to [1, J]$ *satisfies* $a_{\pi_m(1)} q_m\left(\pi_m(1)\right) \ge a_{\pi_m(2)} q_m\left(\pi_m(2)\right) \ge \cdots \ge a_{\pi_m(J)} q_m\left(\pi_m(J)\right)$.

Proof. From (15), (16) and (17), we have

$$\bar{\omega}_j = \bar{b}_j^{(m)} \left[a_j q_m(j) - \bar{\nu} \right], \tag{19}$$

which, when inserted into (16), gives

$$\bar{b}_j^{(m)} \left(\bar{b}_j^{(m)} - 1 \right) \left[a_j q_m(j) - \bar{\nu} \right] = 0. \tag{20}$$

If $\bar{\nu} < a_j q_m(j)$, we have

$$\bar{\omega}_j = \bar{\eta}_j + a_j q_m(j) - \bar{\nu} > 0.$$

Thus, from (16), we have $\bar{b}_j^{(m)} = 1$. Similarly, if $\bar{\nu} \geq a_j q_m(j)$, we have

$$\bar{\eta}_j = \bar{\omega}_j + \bar{\nu} - a_j q_m(j) > 0.$$

Hence, from (15), we have $\bar{b}_j^{(m)} = 0$.

For notational convenience we introduce the functions $\psi_j \colon \mathbb{R} \to [0,1]$, $j = 1, \ldots, J$ as follows

$$\psi_j(\nu) = \begin{cases} 1, & \text{if } \nu < a_j q_m(j) \\ 0, & \text{if } \nu \geq a_j q_m(j). \end{cases} \tag{21}$$

We also define $\psi : \mathbb{R} \to [0, K]$, where $\psi(\nu) = \sum_{j=1}^{J} \psi_j(\nu)$. Note that $\psi(\nu) = K$ for $\nu \in \left(-\infty, a_j q_m(j)\right)$, and $\psi(\nu) = 0$ for $\nu \in \left[a_j q_m(j), \infty\right)$.

It is possible to check for all possible combinations from the file set $[1, J]$ to $[1, J]$ to confirm if the condition given in (21) is satisfied. In order to satisfy the capacity constraint (11) the above solution is guaranteed to exist. The proof is completed by validating that with the strategy above $\bar{\nu}$ is satisfying $\psi(\bar{\nu}) = K$. □

The intuition behind Theorem 2 is that we order the files according to a measure $a_j q_m(j)$, $j = 1, \ldots, J$, where we recall from (6) that

$$q_m(j) = \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}). \tag{22}$$

The measure $a_j q_m(j)$ includes file popularity a_j and also takes into account if neighbours are already storing file j through $q_m(j)$. The factor $q_m(j)$ takes into account the area of overlap in the coverage region with the neighbours through p_s. After ordering the files we store the K 'most popular' files according to the measure $a_j q_m(j)$.

Next we demonstrate that Theorem 2 provides an optimal solution to Problem 2.

Theorem 3. *The optimal solution given in Theorem 2 is a solution to Problem 2.*

Proof. We applied relaxation on Problem 2 and presented the convex, and in fact the linear version of it by Problem 3. With this relaxation we provide allowance on $b_j^{(m)}$ values to take values between the interval $[0, 1]$ instead of simply taking values from the set $\{0, 1\}$. Having solved the problem, the optimal solution given in Theorem 2 provided a combinatorial structure on $b_j^{(m)}$, hence the solution also applies to Problem 2. □

Since our best response update can be solved as a convex and linear optimization problem, it can be done in polynomial time (see e.g., [2],[36]).

Furthermore, following the approach in [5] we can show that ϵ-Nash equilibrium can be achieved relatively fast. An ϵ-Nash equilibrium is characterized by

$$f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)}) \leq f^{(m)}(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)}) - \epsilon, \quad \forall m, \tilde{\mathbf{b}}^{(m)}.$$

At each improvement we aim to decrease the potential function by at least ϵ. If no player can make a move decreasing the potential by at least ϵ we stop and the reached profile corresponds to the ϵ-Nash equilibrium.

Then, it will take no more than $1/\epsilon$ to reach the ϵ-Nash equilibrium. In particular, if we set the value of ϵ less or equal to the minimal improvement value provided in Lemma 4, we actually reach the exact Nash equilibrium.

4.3 Structure of Nash equilibria

In this subsection we provide insight into the structure of the Nash equilibria of the content placement game. We know from the previous subsection that this game is a potential game. The Nash equilibria, therefore, correspond to placement strategies $\mathbf{B}$ that satisfy the Karush-Kuhn-Tucker conditions of Problem 1.

The next result demonstrates that the optimal solution of the relaxed problem follows a threshold strategy for each cache. First all files are ordered according to a function of the placement policies of the neighbouring caches and then the first K files are stored.

Theorem 4. *Let $\bar{\mathbf{B}}$ denote a content placement strategy at a Nash equilibrium of the content placement game. Then*

$$\bar{b}_j^{(m)} = \begin{cases} 1, & \text{if } \bar{\pi}_m^{-1}(j) \le K, \\ 0, & \text{if } \bar{\pi}_m^{-1}(j) > K, \end{cases} \tag{23}$$

where $\bar{\pi}_m : [1, J] \to [1, J]$ satisfies

$$a_{\bar{\pi}_m(1)} \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - \bar{b}^{(\ell)}_{\bar{\pi}_m(1)}) \ge a_{\bar{\pi}_m(2)} \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - \bar{b}^{(\ell)}_{\bar{\pi}_m(2)})$$
$$\ge \cdots \ge a_{\bar{\pi}_m(J)} \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - \bar{b}^{(\ell)}_{\bar{\pi}_m(J)}), \quad \forall m = 1, \ldots, N.$$

Proof. The proof is similar to the proof of Theorem 2, where in this case (32) must hold for all $1 \le m \le N$ simultaneously. The detailed analysis is omitted due to space restrictions. □

4.4 Complexity analysis

In this subsection we provide a bound on the number of iterations that is required to converge to a Nash equilibrium of the content placement game. Also, we provide a bound on the computational complexity of each iteration.

Let us consider *discrete placement* of caches in the plane. More precisely, the locations of caches are restricted to $d\mathbb{Z} = \{(i_1 d, i_2 d) \mid i_1, i_2 \in \mathbb{Z}\}$, where d is the minimum possible distance between caches. Also, the coverage area of each cache is assumed to be the disk of radius r around the location of the cache. The assumption of discrete placement is not restrictive and will, in fact, be satisfied in practical scenarii where the location of base stations is specified in, for instance, whole meters and not with arbitrary precision.

Since we are interested in the complexity and the convergence rate of our algorithm as a function of the network size N and of the library size J we need to consider a sequence (indexed by J) of file popularity distributions. In general it is possible that $a_i \to 0$ as $J \to \infty$. We will assume that for all i, a_i decreases at most polynomially fast in J and say that such a sequence of distributions *scales polynomially*. This condition is satisfied for all practical scenarii, like Zipf distributions. In fact, if the Zipf scaling parameter $\gamma > 1$, then a_i converges to a positive value for all i. In this subsection we assume that the caches are scheduled in round-robin fashion.

In general, there are examples of potential games where converging to a Nash equilibrium by the best response dynamics can take exponential time, see *e.g.*, [3]. In the remainder of this subsection we will show that under discrete placement of caches and polynomial scaling of the popularity distribution, the convergence time is at most polynomial in N and J. Our proof relies on the following result, the proof of which is given in Appendix A.

THEOREM 5. *Let* $\mathbf{B}$ *and* $\tilde{\mathbf{B}}$ *denote the placement before and after one local update, respectively. Consider a discrete placement of caches and polynomial scaling of file popularities. Then*

$$f(\mathbf{B}) \neq f(\tilde{\mathbf{B}}) \Longrightarrow f(\mathbf{B}) - f(\tilde{\mathbf{B}}) \geq \kappa_1 N^{-1} J^{-\kappa_2}, \tag{24}$$

where $\kappa_1 > 0$ *and* $\kappa_2 \geq 0$ *are constants.*

The main result of this subsection is as follows.

THEOREM 6. *Consider discrete placement of caches, polynomial scaling of file popularities and round-robin scheduling of caches. Then, the best response dynamics of the content placement game converges to a Nash equilibrium in at most* $\kappa_1^{-1} N^2 J^{\kappa_2}$ *iterations, with* κ_1 *and* κ_2 *as in Theorem 5.*

PROOF. If we improve the miss probability by making a local update we improve it by at least $\kappa_1 N^{-1} J^{-\kappa_2}$ by Theorem 5. We can make at most $\kappa_1^{-1} N J^{\kappa_2}$ such improvements, because we are minimizing the miss probability, which is bounded between 0 and 1. Furthermore, we cannot have more than $N - 2$ sequential updates in which we are not improving, because we are using a round-robin schedule and not being able to provide a strictly better response for any of the caches implies that we have reached a Nash equilibrium. □

Thus, the complexity of our basic algorithm in the context of discrete placement is polynomial in time. We note that this is quite interesting result as in general the best response dynamics in potential games does not have polynomial time complexity.

Next, we demonstrate that the computational complexity of each update does not increase with the network size or the catalog size.

THEOREM 7. *Consider discrete placement of caches. Then the complexity of each update is constant in both the network size* N *and the catalog size* J*.*

PROOF. In a discrete placement each cache will have at most $(\lceil 2r/d \rceil + 1)^2$ neighbours (including itself). Together, these caches can store at most $K(\lceil 2r/d \rceil + 1)^2$ files. In order to minimize the miss probability, the files that need to be cached will be a subset of the $K(\lceil 2r/d \rceil + 1)^2$ most popular files and there is no need to consider other files in the content library. Therefore, the complexity is independent of the library size. Also, the number of neigbours is independent of N. □

We have presented our complexity result for a discrete placement of caches, but the result can easily be generalized to any placement of caches in which the number of neighbours of caches is bounded. Furthermore, it is indicated in the proof that in each update we only need to consider the $(M + 1)K$ most popular files, where M is the number of neighbours of the caches that is being updated. This result can be strengthened as follows. Once we perform an update for cache m, we know the placement strategies of all its neighbours. It is easy check for the least popular file stored among all neighbours. We denote the index of this least popular file by S_m. Then, one can see that we need to search over first $S_m + K$ files only, where we note that $S_m + K \leq K \times (M + 1)$.

Finally, note that if we relax the assumption of discrete placement and consider arbitrary (continuous) placement of caches, our game will still be in the PLS complexity class [47].

5 SIMULATED ANNEALING APPROACH TO GLOBAL OPTIMALITY

In this section we will first give an example of a network where the best response dynamics converges to a local optimum. Next we will present two potential remedies; stochastic simulated annealing and deterministic simulated annealing, in order to achieve global optimum for such networks.

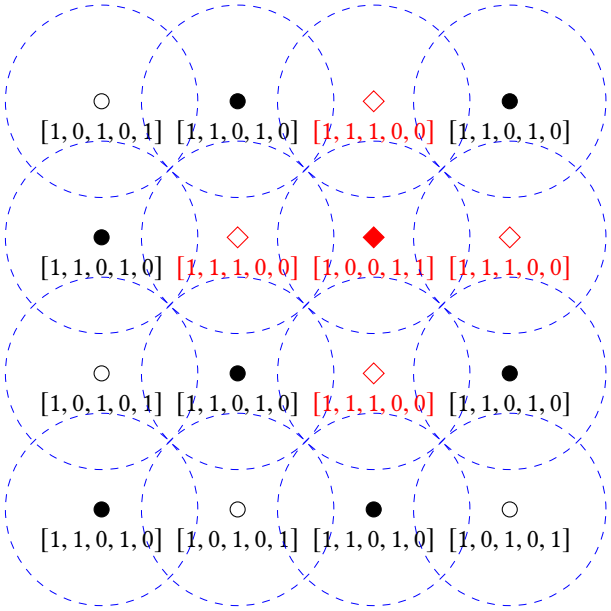


Fig. 1. A final non-optimal file placement strategy example for a 4×4 grid network.

5.1 Best response and local optima

The main aim of this section is to show that the content placement game might get stuck at a local optimum in a symmetric network topology. Caches are located on a 4×4 grid. Caches have three-dimensional coverage area, resulting each cache being located at the center of a torus. An example is shown in Figure 1, each node representing the location of a cache with a toroidal coverage area. Caches are sharing a common coverage area with their neighbours and caches located at the edge of the grid network are neighbouring with the ones located at the opposite edge of the network (depending on the coverage radius.).

Consider the case of caches with K-slot cache memory and the content library of size $J = 1000$. We set the cache capacity of the caches as $K = 3$. We assume a Zipf distribution for the file popularities, setting $\gamma = 1$ and taking a_j according to (1). The coverage radius is set to $r = 700$ m. Distance between caches is set to $d = r\sqrt{2}$ m.

In Figure 1, we have depicted a file placement strategy (only for the first 5 files, the rest are all-zero) that is a Nash equilibrium of the potential content placement game. We will argue below that this Nash equilibrium has a hit probability that is a slightly lower value than the global optimum. Each node is representing a cache. With this strategy, you can verify that KKT conditions are satisfied. It is clear that the red diamond shaped caches are storing different set of files compared to black circle shaped caches. One can also verify that KKT conditions will be satisfied if red diamonds follow the same strategy as in black circles (filled diamond follows the filled circle strategy and empty ones follow the empty circle strategy, respectively.). In this case, the hit probability will give the global optimum. Just to give some intuition, consider the following example: c_2 is available in both empty diamonds and filled circles and c_5 is not present in any of them. Hence, the intersecting area between empty diamond and filled circle will have a performance penalty, which reduces the total hit probability. This would not occur if the diamonds were circles.

5.2 Stochastic simulated annealing

In this subsection, following the framework of Hajek [28], we provide a simulated annealing (SA) algorithm that will converge to the global optimum with probability 1. Intuitively, the idea of the algorithm is to allow, with a small probability, for arbitrary changes to the placement policy at a cache during a local update.

More formally, we will construct a discrete-time Markov chain $\mathbf{B}_0, \mathbf{B}_1, \ldots$, that converges to the optimal placement w.p. 1. To that end, for a state $\mathbf{B}$ we define a neighbourhood $\mathcal{N}(\mathbf{B})$ as those placement policies that differ from $\mathbf{B}$ in at most one column, i.e. that differ in at most one cache. Within a column, any change that satisfies the capacity constraint $\sum_{j=1}^{J} b_j^{(m)} = K$ is allowed.

Given $\mathbf{B}_t = \mathbf{B}$, a potential next state $\mathbf{Y}_t$ is chosen from $\mathcal{N}(\mathbf{B})$ with probability distribution $P[\mathbf{Y}_t = \tilde{\mathbf{B}} | \mathbf{B}_t = \mathbf{B}] = R(\mathbf{B}, \tilde{\mathbf{B}})$, defined as

$$R(\mathbf{B}, \tilde{\mathbf{B}}) = \frac{1}{N} \sum_{m=1}^{N} R_m(\mathbf{B}, \tilde{\mathbf{B}}), \tag{25}$$

with

$$R_m(\mathbf{B}, \tilde{\mathbf{B}}) = \begin{cases} \tilde{p}, & \text{if } \tilde{b}^{(m)} = \bar{b}^{(m)},\ \tilde{b}^{(-m)} = b^{(-m)}, \\ \frac{1-\tilde{p}}{\binom{J}{K}-1}, & \text{if } \tilde{b}^{(m)} \neq \bar{b}^{(m)},\ \sum_{j=1}^{J} \tilde{b}_j^{(m)} = K,\ \tilde{b}^{(-m)} = b^{(-m)}, \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

where $0 < \tilde{p} < 1$ is a constant and where $\bar{b}^{(m)}$ is a solution to Problem 2, i.e. $\bar{b}^{(m)}$ is the best response for cache m in the content placement game. Then, we set

$$\mathbf{B}_{t+1} = \begin{cases} \mathbf{Y}_t, & \text{with probability } \hat{p}_t, \\ \mathbf{B}_t, & \text{otherwise,} \end{cases} \tag{27}$$

where

$$\hat{p}_t = \exp\left[\frac{-\max\{f(\mathbf{Y}_t) - f(\mathbf{B}), 0\}}{T_t} \right], \tag{28}$$

with

$$T_t = \frac{d}{\log(t+1)}, \tag{29}$$

where $d > 0$ is a constant.

Theorem 8. *If $d \geq 1$, then Markov chain $\{B_t\}$ converges with probability 1 to a global optimum of Problem 1.*

Proof. This follows directly from [28, Theorem 1], for which we demonstrate here that we satisfy all conditions. First, $\{B_t\}$ is irreducible. Second, we have the property that

$$\tilde{\mathbf{B}} \in \mathcal{N}(\mathbf{B}) \iff \mathbf{B} \in \mathcal{N}(\tilde{\mathbf{B}}), \tag{30}$$

for all $\mathbf{B}$ and $\tilde{\mathbf{B}}$. It can be readily verified that (30) is sufficient for weak reversibility as defined in [28]. Finally, since our objective function is a probability, it is bounded between 0 and 1. The depth of a local minimum is, therefore, at most 1 and it is sufficient to consider $d \geq 1$. □

Note that the selection of $\tilde{b}^{(m)} \neq \bar{b}^{(m)}$ in (26) uniformly at random requires the computation of $\binom{J}{K}$. We use Fisher-Yates shuffle [17] to obtain the random permutation over J files and store the first K many files after shuffling to produce random $\tilde{b}^{(m)}$ vectors. With Durstenfeld's extended Fisher-Yates shuffle algorithm [15], computational complexity is $O(J)$.

Algorithm 1: Stochastic simulated annealing (SSA)

initialize $\mathbf{b}^{(m)} = [\underbrace{1,\ldots,1}_{K \text{ many}},0,\ldots,0]$, $\forall m \in \{1,\ldots,N\}$;
for $t = 1, 2, \ldots$ **do**
  m = Uniform(N);
  Set the temperature T_t using (29);
  Pick a random number $\rho \in [0,1]$;
  if $\rho < \tilde{p}$ **then**
    Solve Problem 2 for cache m and find $\bar{\mathbf{b}}^{(m)}$ using the information coming from neighbours;
    Set $\tilde{\mathbf{b}}^{(m)} = \bar{\mathbf{b}}^{(m)}$
  else
    Set $\tilde{\mathbf{b}}^{(m)}$ = Uniformly at random;
  end
  Update $\tilde{\mathbf{B}}_k$ by inserting the designated row;
  Compute $\hat{p}_t$ using (28);
  Pick a random number $\mu \in [0,1]$;
  if $\mu < \hat{p}_t$ **then**
    Set $\mathbf{B}_{t+1} = \tilde{\mathbf{B}}_t$
  else
    Set $\mathbf{B}_{t+1} = \mathbf{B}_t$
  end
end

5.3 Deterministic simulated annealing

In this subsection we will briefly present another new algorithm based on deterministic simulated annealing. The basic idea of the deterministic simulated annealing or homotopy approach (see e.g., [39],[45]) is to gradually transform an easier problem to the original, more difficult, problem. In deterministic simulated annealing (DSA), an initial τ is set. The problem formulation is similar to the procedure given in Section 4.2. We have the same problem as in Problem 3 with modified boundary constraints for the file placement probabilities, *i.e.*, $b_j^{(m)}$ can now take values from the closed set $[\tau, 1-\tau]$ instead of the closed set $[0,1]$. Following a similar analysis in the aforementioned section, we have the following theorem, which we state without proof.

Theorem 9. *The optimal solution to DSA problem is given by*

$$\bar{b}_j^{(m)} = \begin{cases} 1-\tau, & \text{if } \pi_m^{-1}(j) < \lceil K - \tau J \rceil + 1 \\ \delta, & \text{if } \pi_m^{-1}(j) = \lceil K - \tau J \rceil + 1 \\ \tau, & \text{if } \pi_m^{-1}(j) > \lceil K - \tau J \rceil + 1, \end{cases} \tag{31}$$

where $\pi_m : [1,J] \to [1,J]$ *satisfies* $a_{\pi_m(1)} q_m(\pi_m(1)) \geq a_{\pi_m(2)} q_m(\pi_m(2)) \geq \cdots \geq a_{\pi_m(J)} q_m(\pi_m(J))$, *and* $\delta = K - [(\lceil K - \tau J \rceil)(1-\tau) + (J - \lceil K - \tau J \rceil - 1)\tau]$.

At the end of each iteration step, τ is reduced. The main idea behind the algorithm is to avoid entering a path that in the end KKT conditions are hold but the final hit probability gives a local minimum instead of the global one. The algorithm stops when $f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)})$ converges $\forall m \in \{1,\ldots,N\}$, *i.e.*, a full round over all caches

$\{1, \ldots, N\}$ does not give an improvement in hit probability. DSA applied to our problem of interest is shown in Algorithm 2.

Algorithm 2: Deterministic simulated annealing (DSA)

initialize $\mathbf{b}^{(m)} = [0, \ldots, 0]$, $\forall m \in \{1, \ldots, N\}$;
set $imp(m) = 1$, $\forall m \in \{1, \ldots, N\}$;
set **imp** $= [imp(1), \ldots, imp(N)]$;
set the initial τ;
while imp $\neq$ **0 do**
  m = Uniform(N);
  Set $imp(m) = 0$;
  Solve Problem 2 for $b_j^{(m)} \in [\tau, 1-\tau]$ for cache m and find $\bar{\mathbf{b}}^{(m)}$ using the information coming from neighbours;
  Compute $f^{(m)}(\bar{\mathbf{b}}^{(\mathbf{m})}, \mathbf{b}^{(-m)})$;
  if $f^{(m)}(\bar{\mathbf{b}}^{(\mathbf{m})}, \mathbf{b}^{(-m)}) - f^{(m)}(\mathbf{b}^{(\mathbf{m})}, \mathbf{b}^{(-m)}) \neq 0$ **then**
    $imp(m) = 1$
  end
  Decrease τ.
end

Corollary 1. *The optimal solution given in Theorem 9 is a solution to Problem 2 when each $b_j^{(m)}$ is rounded to its nearest integer as $\tau \to 0$.*

Proof. We already showed in Theorem 3 that the solution of Problem 3 is a solution to Problem 2. With the boundary conditions presented in DSA, we provide allowance on $b_j^{(m)}$ values to take values between the interval $[\tau, 1-\tau]$ instead of taking values from the set $[0, 1]$. Having solved the problem, the optimal solution given in Theorem 9 is an equivalent to the one given in Theorem 2 as $\tau \to 0$. Hence, when τ is small enough, rounding $b_j^{(m)}$s to their nearest integers gives the solution to Problem 2. □

We will see from numerical results in Section 6 that DSA is an interesting approach to escape from the local optimum for the grid network. Considering the same cache update sequence causing the diamond shaped strategies illustrated in Figure 1, optimizing the caches in the same sequence with DSA prevents the system ending up with diamond shaped strategies at the caches and the hit probability converges to the global optimum.

6 NUMERICAL EVALUATION

In this section we will present practical implementations of our algorithms for the content placement game and evaluate our theoretical results according to a network of caches with their geographical locations following a homogeneous Poisson process, a real wireless network and a grid network. Finally we will present an example illustrating the resulting placement strategies of the caches obtained by our algorithms.

6.1 The ROBR and RRBR algorithms

The basic idea of our algorithm (which comes in two variants, RRBR and ROBR) is to repeatedly perform best response dynamics presented in Section 4.2. We introduce some notation in the next definition.

Applying distributed optimization to cache m gives the new placement policy denoted by $\bar{\mathbf{b}}^{(m)}$ which is given by Theorem 2. Hence,

$$\bar{b}_j^{(m)} = \begin{cases} 1, & \text{if } \pi_m^{-1}(j) \leq K, \\ 0, & \text{if } \pi_m^{-1}(j) > K, \end{cases} \tag{32}$$

where $\pi_m : [1, J] \rightarrow [1, J]$ satisfying $a_{\pi_m(1)} q_m(\pi_m(1)) \geq a_{\pi_m(2)} q_m(\pi_m(2)) \geq \cdots \geq a_{\pi_m(J)} q_m(\pi_m(J))$.

As neighbouring caches share information with each other, the idea is to see if applying distributed optimization iteratively and updating the file placement strategies over all caches gives $\bar{\mathbf{b}}^{(m)}$ for all $m \in [1 : N]$ yielding to the global optimum given in Theorem 4. To check this, we define the following algorithms.

For Round-Robin Best Response (RRBR) algorithm, we update the caches following the sequence of the indices of the caches. We assume that all caches are initially storing the most popular K files. The algorithm stops when $f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)})$ converges $\forall m \in \{1, \ldots, N\}$, *i.e.*, a full round over all caches $\{1, \ldots, N\}$ does not give an improvement in hit probability. RRBR algorithm is shown in Algorithm 3.

Algorithm 3: Round-Robin Best Response (RRBR)

initialize $\mathbf{b}^{(m)} = [\underbrace{1, \ldots, 1}_{K \text{ many}}, 0, \ldots, 0]$, $\forall m \in \{1, \ldots, N\}$;
set imp = 1;
while *imp = 1* **do**
  Set *imp* = 0;
  for $m = 1 : N$ **do**
    Solve Problem 2 for cache m and find $\bar{\mathbf{b}}^{(m)}$ using the information coming from neighbours;
    Compute $f^{(m)}(\bar{\mathbf{b}}^{(\mathbf{m})}, \mathbf{b}^{(-m)})$;
    if $f^{(m)}(\bar{\mathbf{b}}^{(\mathbf{m})}, \mathbf{b}^{(-m)}) - f^{(m)}(\mathbf{b}^{(\mathbf{m})}, \mathbf{b}^{(-m)}) \neq 0$ **then**
      imp = 1
    end
  end
end

It is also possible to update the caches by following a random selection algorithm. For Random Order Best Response (ROBR) Algorithm, at every iteration step, a random cache is chosen uniformly from the total cache set $\{1, \ldots, N\}$ and updated. We assume that all caches are initially storing the most popular K files. The algorithm stops when $f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)})$ converges $\forall m \in \{1, \ldots, N\}$, *i.e.*, a full round over all caches $\{1, \ldots, N\}$ does not give an improvement in hit probability. ROBR algorithm is shown in Algorithm 4.

6.2 Poisson placement of caches

Consider the case of caches with K-slot cache memory and the content library of size $J = 100$. We set $K = 3$. We assume a Zipf distribution for the file popularities, setting $\gamma = 1$ and taking a_j according to (1). We have chosen the intensity of the Poisson process equal to $\lambda = 8 \times 10^{-6}$. Base stations' coverage radius is set to $r = 1000$ m.

For the stochastic simulated annealing, two different cooling schedules are considered, *i.e.*, two different d values.

Our goals are to see if the proposed solution algorithms converge, to compare the performances of the algorithms and to compare them with the probabilistic placement strategy [8] and with multi-LRU-One caching [24].

Algorithm 4: Random Order Best Response (ROBR)

initialize $\mathbf{b}^{(m)} = [\underbrace{1,\ldots,1}_{K \text{ many}},0,\ldots,0]$, $\forall m \in \{1,\ldots,N\}$;
set $imp(m) = 1, \forall m \in \{1,\ldots,N\}$;
set **imp** $= [imp(1),\ldots,imp(N)]$;
while imp $\neq$ **0 do**
  m = Uniform(N);
  Set $imp(m) = 0$;
  Solve Problem 2 for cache m and find $\bar{\mathbf{b}}^{(m)}$ using the information coming from neighbours;
  Compute $f^{(m)}(\bar{\mathbf{b}}^{(\mathbf{m})}, \mathbf{b}^{(-m)})$;
  if $f^{(m)}(\bar{\mathbf{b}}^{(\mathbf{m})}, \mathbf{b}^{(-m)}) - f^{(m)}(\mathbf{b}^{(\mathbf{m})}, \mathbf{b}^{(-m)}) \neq 0$ **then**
    $imp(m) = 1$
  end
end

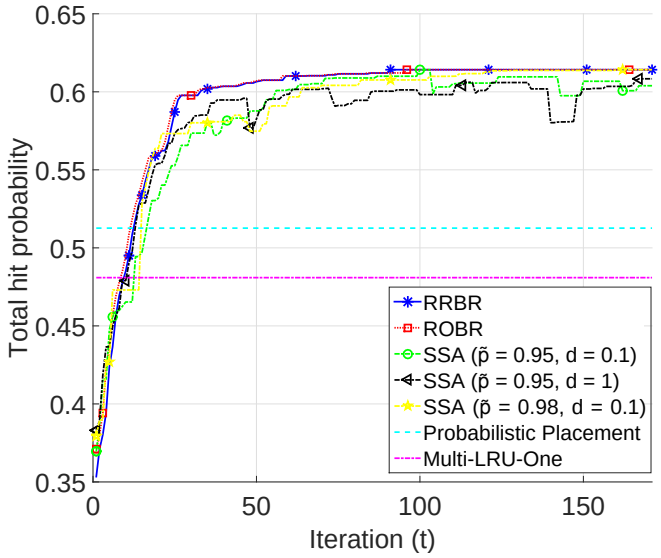


Fig. 2. Hit probability evolution for different algorithms ($J = 100, K = 3$).

In [8], it has been already shown that it is not optimal to cache the most popular contents everywhere. In Multi-LRU caching policies [24], the main assumption is that a user who is covered by multiple caches can check all the caches for the requested file and download it from any one that has it in its inventory. In Multi-LRU-One caching policy, if the requested file is found in a non-empty subset of the caches that is covering a user, only one cache from the subset is updated. If the object is not found in any cache, it is inserted only in one. In this work, the selected cache for the update will be picked uniformly at random from the caches covering the user.

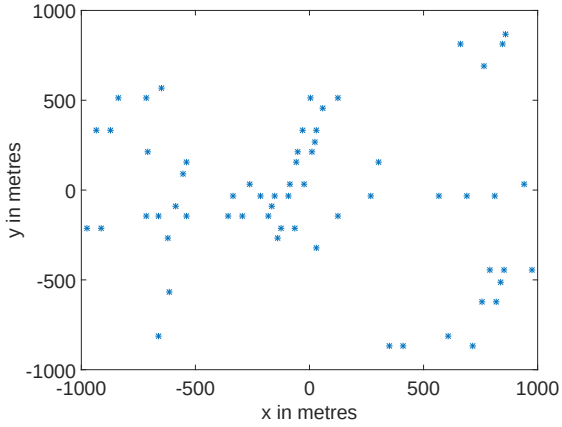


Fig. 3. Location of Base Stations from OpenMobileNetwork dataset.

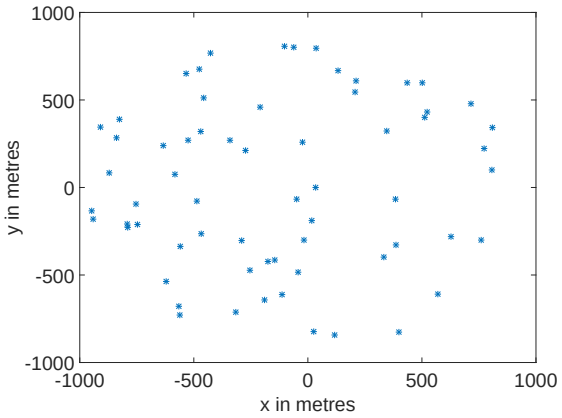


Fig. 4. A realization of the Spatial Homogeneous Poisson Process.

In Figure 2, the hit probability (one-minus-miss probability) evolution for the proposed solution algorithms are shown. Both RRBR, ROBR and SSA algorithms converge to the same total hit probability value. First, we ran a simulation with $d = 1$, because by Theorem 8 this guarantees convergence. In all our experiments, we have observed that random selections of the SSA decreases the hit probability by no more than 0.1 until convergence. Consequently, we ran another SSA with $d = 0.1$. Both SSA algorithms converge to the same value, however using a smaller d increases the convergence speed of the SSA. This is not surprising because the temperature is decreased further when d is smaller. Also, it is easy to observe that using a larger $\tilde{p}$ helps the algorithm stay closer to the optimal solution at each iteration step.

We see that coordination between caches in ROBR and RRBR helps to improve the overall performance compared to probabilistic placement proposed in [8] and Multi-LRU-One decentralized caching policy proposed in [24]. We observe significant increase in total hit probability when we exploit the information sharing between neighbouring caches.

6.3 A real wireless network: Berlin network

In this section we will evaluate the performance of the topology of a real wireless network. We have taken the positions of 3G base stations provided by the OpenMobileNetwork project [50]. The base stations are situated in the area 1.95×1.74 kms around the TU-Berlin campus. Base stations' coverage radius is equal to $r = 700$ m. The positions of the base stations from the OpenMobileNetwork project is shown in Figure 3. We note that the base stations of the real network are more clustered because they are typically situated along the roads.

In order to compare the Spatial Homogeneous Poisson Process performance with the real network, we have chosen the intensity of the Poisson process equal to the density of base stations in the real network. In Figure 4 one can see a realization of the Spatial Homogeneous Poisson Process with $\lambda = 1.8324 \times 10^{-5}$. The density is small because we measure distances in metres. Base stations' coverage radius is set to $r = 700$ m. We have averaged over 100 realizations of the Poisson process.

Consider the case of caches with K-slot cache memory and the content library of size $J = 200$. We set $K = 3$. We assume a Zipf distribution for the file popularities, setting $\gamma = 1$ and taking a_j according to (1). We find the probabilistic placement strategy by the strategy given in [8] by using the parameters given earlier ($\lambda = 1.8324 \times 10^{-5}$ and $r = 700$), store the files accordingly over all caches and compute the hit probability for

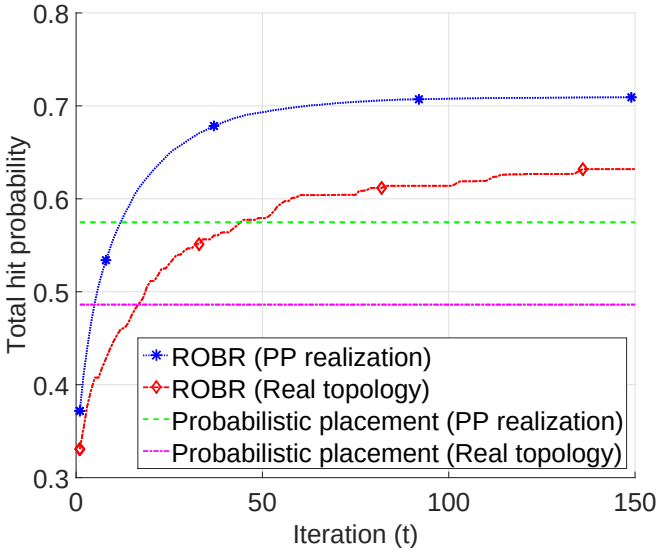


Fig. 5. Hit probability evolution ($J = 200, K = 3$).

the probabilistic placement policy. In Figure 5, the hit probability (one-minus-miss probability) evolution for the ROBR algorithm and the probabilistic placement policy for both averaged over 100 Poisson Point realizations and real OpenMobileNetwork topology is shown. We see that running the ROBR algorithm on both homogeneous Poisson Process realizations and the real topology performs significantly better than the probabilistic placement policy. We observe that the total hit probability for the average of 100 homogeneous Poisson Process realizations is higher than the one of real topology. The reason is that the spread fashion of the homogeneous Poisson Point realizations allows more coordination between the base stations compared to the clustered fashion of the real topology, leading to the fact that they share more information between each other at each iteration leading to higher hit probability in the end. Note that in real topology shown in Figure 3, there are many uncovered areas; however we only consider the regions that are covered by at least one of the base stations while computing our performance metric.

Now we will present a more realistic example in terms of library size. We can get the numerical results for huge library size by using ROBR Algorithm. Consider the case of caches with K-slot cache memory and the content library of size $J = 100.000$. We set $K = 10$. We assume a Zipf distribution for the file popularities, setting $\gamma = 1$ and taking a_j according to (1). We find the probabilistic placement strategy by the strategy given in [8] by using the parameters given earlier ($\lambda = 1.8324 \times 10^{-5}$ and $r = 700$), store the files accordingly over all caches and compute the hit probability for the probabilistic placement policy. We store the most popular K files over all caches in most popular content placement algorithm. In Figure 6, the hit probability (one-minus-miss probability) evolution for the ROBR Algorithm for real OpenMobileNetwork topology is shown. Recalling that there are $N = 62$ caches in total and all caches have capacity $K = 10$, all caches in the network can only store first $K \times N = 620$ files out of 100.000. Then from (1), it is trivial to see that 42.04% of the files will always be missed. Due to nonhomogeneous scattering of the base station locations in real topology, storing the most popular content gives slight performance advantage over the probabilistic placement policy. We see that running the ROBR algorithm on the real topology performs significantly better than the probabilistic placement policy. The clustered fashion of the real network

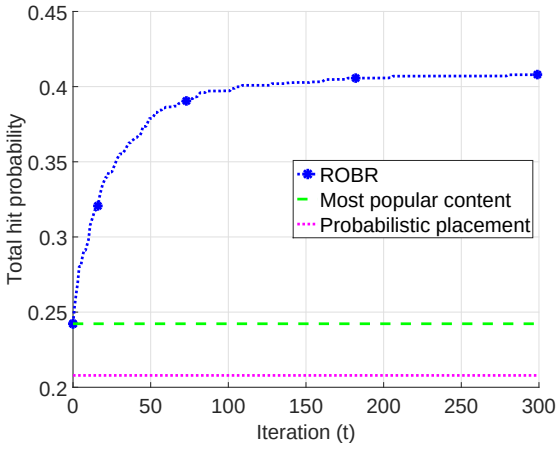


Fig. 6. Hit probability evolution ($J = 100000, K = 10$).

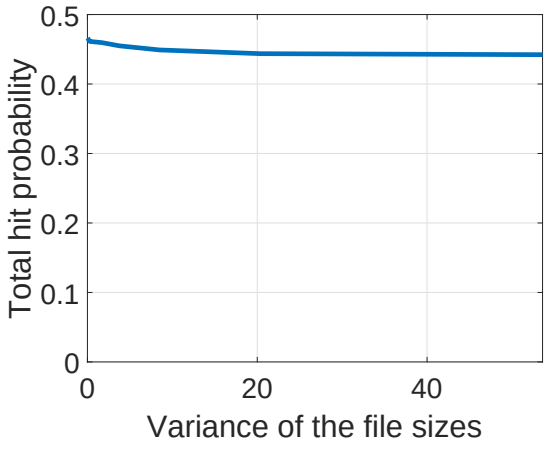


Fig. 7. Hit probability evolution for nonhomogeneous file sizes ($J = 100000, K = 20$).

topology allows more coordination between the base stations, leading to the fact that they share more information between each other at each iteration leading to higher hit probability. Also note that in real topology shown in Figure 3, there are many uncovered areas; however we only consider the regions that are covered by at least one of the base stations while computing our performance metric.

Now we will present the performance of our algorithm when the file sizes are nonhomogeneous. We will present the numerical results again by using ROBR Algorithm. Consider the case of caches with K-slot cache memory and the content library of size $J = 100.000$. We set $K = 20$. We assume a Zipf distribution for the file popularities, setting $\gamma = 1$ and taking a_j according to (1).

For every file a_j we generate a random number ζ_j from a log-normal distribution with mean 1 for various variances. Then the file a_j has size ζ_j. Next, we define how the cache placement strategy works. After running the ROBR algorithm, depending on the resulting placement strategy obtained from the best response dynamics, each cache can store a file unless the sum of the size of the stored files exceed the cache capacity, *i.e.*, the cache starts filling its capacity by the file that it is supposed to store with the lowest index until it saturates (and skip the ones in between if they exceed the capacity and continue with the next one.).

In Figure 7, the converged hit probability (one-minus-miss probability) values for the ROBR Algorithm for real OpenMobileNetwork topology for nonhomogeneous file sizes is shown. Recalling that there are $N = 62$ caches in total and all caches have capacity $K = 20$, all caches in the network can only store first $K \times N = 620$ files out of 100.000. Then from (1), it is trivial to see that 36.31% of the files will always be missed. As the variance between file sizes increases, the ROBR algorithm converges to smaller hit probability. However, the difference is very small and negligible. If the most popular files have huge capacity, this can also be solved with adjusting the cache capacities accordingly.

6.4 Grid network

In this section we will present the performance analysis of a 4×4 grid network. The main aim of this section is to show that ROBR algorithm might get stuck at a local optimum in a symmetric network topology.

Consider the case of caches with K-slot cache memory and the content library of size $J = 1000$. We set the cache capacity of the caches as $K = 3$. We assume a Zipf distribution for the file popularities, setting $\gamma = 1$ and

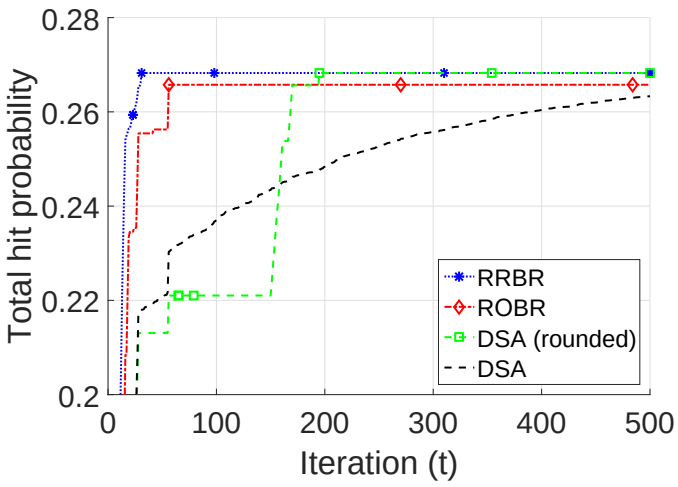


Fig. 8. Hit probability evolution for 4×4 grid network.

taking a_j according to (1). Base stations' coverage radius is set to $r = 700$ m. Distance between caches is set to $d = r\sqrt{2}$ m. For DSA, initial τ is set to $\tau = 10^{-3}$ and decreased exponentially, resulting in having $\tau = 10^{-6}$ at the 1500th iteration step.

In Figure 8, the hit probability evolution for 4×4 grid network is shown. In previous network topology illustrations, *e.g.*, in spatial homogeneous Poisson point process example or Berlin network, we have a non-symmetric geographical cache placement and we have observed that the hit probability converges to the global optimum with ROBR and RRBR (as it converges to the same value with SA.). However, in this example we observe that the hit probability may converge to slightly different hit probabilities (the difference is in the order of 10^{-3}.) by running ROBR algorithm multiple times. We see that DSA also converges to the global optimum rapidly, when each $b_j^{(m)}$ is rounded to its nearest integer, verifying Corollary 1 numerically.

6.5 Resulting placement strategy: An example

In this section we will present a simple example showing the resulting placement strategies of the caches obtained by the ROBR algorithm. For this specific example, all three caches have K-slot cache memory and the content library of size $J = 200$. We set $K = 3$. We assume a Zipf distribution for the file popularities, setting $\gamma = 1$ and taking a_j according to (1).

It can be seen in Figure 9 that as the common coverage area between two neighbouring caches increases, the probability of storing less popular files increases. In this specific scenario, the blue cache has been updated first and is always storing the three most popular files. As the neighbouring area between itself and red cache increases, red cache starts storing less popular files. It is evident that no simple heuristic can be used to give a similar placement strategy covering all such scenarios. Combining the structure of the resulting placement strategies with the performance evaluations that we have presented earlier, we conclude that our low-complexity distributed algorithm provides significant hit-probability improvement by exploiting the information sharing between neighbouring caches.

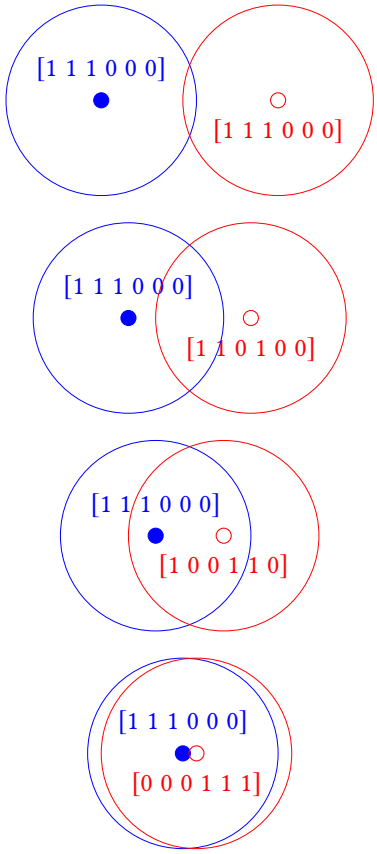


Fig. 9. An example of the resulting optimal placement strategies for a small network.

7 DISCUSSION AND CONCLUSION

In the current paper we have provided a low-complexity asynchronously distributed cooperative caching algorithm in cellular networks when there is communication only between caches with overlapping coverage areas. We have provided a game theoretic perspective on our algorithm and have related the algorithm to a best response dynamics in a game. Using this connection, we have shown that the complexity of each best response step is independent of the catalog size, linear in cache capacity and linear in the maximum number of caches that cover a certain area. Furthermore, we have shown that the overall algorithm complexity for the discrete placement of caches is polynomial in network size and catalog size. Moreover, we have shown that the algorithm converges in just a few iterations by the aid of practical examples. In most cases of interest, our basic low-complexity algorithm finds the best Nash equilibrium corresponding to the global optimum. We have given an upper bound to the rate of convergence of our algorithm by using the value for which we have found for the minimum improvement in hit probability in the overall network. For the cases where the algorithm converges to a local optimum, we have shown that the resulting performance gap in comparison with the global optimum is very small. We have also provided two simulated annealing based extensions of our basic algorithm to find global optimum. We have demonstrated the hit probability evolution on real and synthetic networks and have shown that our distributed cooperative caching algorithm performs significantly better than storing the most popular content, probabilistic

content placement policy and Multi-LRU caching policies. For libraries with nonhomogeneous file sizes, we have shown that as the variance of the log-normal file sizes increases, the total hit probability decreases. However, the difference turns out to be very small. We have also given a practical example where the basic algorithm converges to the local optimum, showed that the performance gap is very small, and our simulated annealing based algorithms converge to the global optimum even when the same cache update sequence of the basic algorithm has been followed. Finally, we have provided an example of the resulting optimal placement strategies for a small network.

In future work we will generalize this analysis for instance to consider time-varying file popularities. In particular, the aim is to obtain a more fundamental insight into the behaviour of more dynamic caching models in stochastic geometry settings.

REFERENCES

[1] E. Altman, K. Avrachenkov, and J. Goseling, "Distributed storage in the plane", *Networking Conference, IFIP 2014*, pp. 1-9, Trondheim, Norway, June 2014.

[2] E. Altman, B. Gaujal, and A. Hordijk, *Discrete-event control of stochastic networks: Multimodularity and regularity*, Springer, 2003.

[3] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden, "The price of stability for network design with fair cost allocation", *SIAM Journal on Computing*, vol. 38, no. 4, pp. 1602-1623, 2008.

[4] K. Avrachenkov, X. Bai, and J. Goseling, "Optimization of caching devices with geometric constraints", in press for *International Journal of Performance Evaluation*, also *ArXiv: 1602.03635*, 2016.

[5] K. Avrachenkov, J. Elias, F. Martignon, G. Neglia, and L. Petrosyan, "Cooperative Network Design: A Nash bargaining solution approach", *Computer Networks*, vol. 83, pp. 265-279, 2015.

[6] I. Baev, R. Rajaraman, and C. Swamy, "Approximation algorithms for data placement problems", *SIAM Journal on Computing*, vol. 38, no.4, pp. 1411-1429, 2008.

[7] E. Bastug, M. Bennis, and M. Debbah, "Cache-enabled small cell networks: Modeling and tradeoffs", *11th International Symposium on Wireless Communications Systems*, pp. 649-653, 2014.

[8] B. Błaszczyszyn, and A. Giovanidis, "Optimal geographic caching in cellular networks", *IEEE International Conference on Communications (ICC) 2015*, pp. 3358-3363, London, UK, June 2015.

[9] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks", *Proceedings of IEEE INFOCOM 2010*.

[10] A. Chattopadhyay, and B. Błaszczyszyn, "Gibbsian on-line distributed content caching strategy for cellular networks", *arXiv preprint arXiv: 1610.02318*, 2016.

[11] W. Chai, D. He, I. Psaras, and G. Pavlou, "Cache "less for more" in information-centric networks", *Proceedings of IFIP Networking 2012*, pp. 27-40.

[12] H. Che, Y. Tung, and Z. Wang, "Hierarchical web caching systems: Modeling, design and experimental results", *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 7, pp. 1305-1314, 2002.

[13] A. Dan, and D. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes", *ACM Performance Evaluation Review*, vol. 18, no. 1, pp. 143-152, 1990.

[14] M. Dehghan, L. Massoulie, D. Towsley, D. Menasche, Y. C. Tay, "A utility optimization approach to network cache design", *Proccedings of IEEE INFOCOM 2016*, San Francisco, CA, USA, April 2016.

[15] R. Durstenfeld, "Algorithm 235: Random permutation", *Communications of the ACM Magazine*, vol.7, no. 7, p. 420, July 1964.

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

[17] A. R. Fisher, and F. Yates, *Statistical tables for biological, agricultural and medical research*, 1st edn. Oliver and Boyd, Edinburgh, 1938.

[18] N.C. Fofack, P. Nain, G. Neglia, and D. Towsley, "Analysis of TTL-based cache networks", *Proccedings of VALUETOOLS 2012*.

[19] N.C. Fofack, M. Dehghan, D. Towsley, M. Badov, and D.L. Goeckel, "On the performance of general cache networks", *Proccedings of VALUETOOLS 2014*.

[20] N.C. Fofack, P. Nain, G. Neglia, and D. Towsley, "Performance evaluation of hierarchical TTL-based cache networks", *Computer Networks*, vol. 65, pp. 212-231, 2014.

[21] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for LRU cache performance", *Proceedings of ITC 2012*.

[22] C. Fricker, P. Robert, J. Roberts, and N. Sbihi, "Impact of traffic mix on caching performance in a content-centric network", *Proccedings of IEEE NOMEN 2012*.

[23] M. Garetto, E. Leonardi, and V. Martina, "A unified approach to the performance analysis of caching systems", *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 1, no. 3, p.12, 2016.

[24] A. Giovanidis, and A. Avranas, "Spatial multi-LRU caching for wireless networks with coverage overlaps", *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*, pp. 403-405, Antibes Juan-les-Pins, France, June 2016. An extended version is available at `https://arxiv.org/abs/1612.04363`
[25] N. Golrezaei, K. Shanmugam, A.G. Dimakis, A.F. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers", *Proceedings of IEEE INFOCOM 2012*.
[26] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and Device-to-Device collaboration: A new architecture for wireless video distribution", *IEEE Communications Magazine*, vol. 51, no. 4, pp. 142-149, April 2013.
[27] J. Goseling, O. Simeone, and P. Popovski, "Delivery latency regions in Fog-RANs with edge caching and cloud processing ", *arXiv preprint arXiv: 1701.06303*, 2017.
[28] B. Hajek, "Cooling schedules for optimal annealing ", *Mathematics of operations research*, vol. 13, no. 2, pp. 311-329, May 1988.
[29] S. Ioannidis, L. Massoulie, A. Chaintreau, "Distributed caching over heterogeneous mobile networks", *SIGMETRICS 2010*, pp. 311-322, NY, USA, June 2010.
[30] W. Jiang, S. Ioannidis, L. Massoulié, and F. Picconi, "Orchestrating massively distributed CDNs", *Proceedings of ACM CoNEXT 2012*. pp. 133-144.
[31] M. A. Maddah-Ali, and U. Niesen, "Fundamental limits of caching", *IEEE Transactions on Information Theory*, vol. 60, pp. 2856 - 2867, May 2014.
[32] M. A. Maddah-Ali, and U. Niesen, "Cache-aided interference channels", *IEEE International Symposium on Information Theory Proceedings (ISIT) 2015*, pp. 809-813, June 2015.
[33] S. Moharir, J. Ghaderi, S. Sanghavi, and S. Shakkottai, "Serving content with unknown demand: the high-dimensional regime", *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 1, pp. 435-447.
[34] D. Monderer, and L. Shapley, "Potential games", *Games and Economic Behavior*, vol. 14, pp. 124-143, 1996.
[35] G. Neglia, D. Carra, and P. Michiardi, "Cache policies for linear utility maximization", *Proceedings of IEEE INFOCOM 2017*.
[36] Y. Nesterov, and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*. SIAM.
[37] M. E. J. Newman, "Power laws, Pareto distributions and Zipf's law", *Contemporary Physics*, vol.46, pp. 323-351, 2005.
[38] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation algorithms for mobile data caching in small cell networks", *IEEE Transactions on Communications*, vol. 62, no.10, pp. 3665-3677, October 2014.
[39] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems", *Proceedings of the IEEE*, vol. 86, no.11, pp. 2210-2239, 1998.
[40] E.J. Rosensweig, J. Kurose, and D. Towsley, "Approximate models for general cache networks". *Proceedings of IEEE INFOCOM 2010*.
[41] E.J. Rosensweig, D.S. Menasche, and J. Kurose, "On the steady-state of cache networks", *Proccedings of IEEE INFOCOM 2013*.
[42] A. Sengupta, R. Tandon, and O. Simeone, "Cloud and cache-aided wireless networks: Fundamental latency trade-offs", *arXiv preprint arXiv: 1605.01690*, 2016.
[43] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: wireless content delivery through distributed caching helpers", *IEEE Transactions on Information Theory*, vol. 59, no.12, pp. 8402-8413, December 2013.
[44] F. Shen, K. Hamidouche, E. Bastug, and M. Debbah, "A Stackelberg game for incentive proactive caching mechanisms in wireless networks", In Proceedings of IEEE GLOBECOM 2016.
[45] V. Sindhwani, S.S. Keerthi, and O. Chapelle, "Deterministic annealing for semi-supervised kernel machines", In Proceedings of ICML 2006, pp. 841-848.
[46] E. Tardos, and T. Wexler, "Network formation games and the potential function method", chapter in *Algorithmic Game Theory*, pp.487-516, 2007.
[47] M. Yannakakis, "Equilibria, fixed points, and complexity classes", *Computer Science Review*, vol. 3, no.2, pp. 71-85, May 2009.
[48] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey", *Computer Networks*, vol. 57, no. 16, pp. 3128-3141, 2013.
[49] J. Zhang, and P. Elia, "Fundamental limits of cache-aided wireless BC: Interplay of coded-caching and CSIT feedback", *IEEE Trans. on Information Theory*, vol. 63, no. 5, 2017.
[50] OpenMobileNetwork, `http://map.openmobilenetwork.org/`.

A PROOF OF THEOREM 5

In this section we provide a lower bound on the improvement in hit probability that is offered after one best response under *discrete placement* of caches. We denote by ϵ the minimum improvement that can be guaranteed in a step.

LEMMA 2. *The minimum improvement ϵ is obtained when there is only a single file update in the cache.*

Proof. The initialization of the algorithms ensures that we have exactly K 1's and $J-K$ 0's in the placement policy for each cache and Theorem 2 ensures that this property is preserved. Therefore, we can decompose each update in the placement policy into single changes (replacing one file by another file). Because the new policy is locally optimal, each such change brings a positive improvement (Otherwise the local optimum would be a policy that is obtained by excluding the negative improvement). The smallest improvement is provided making a single change. □

Lemma 3. *If the hit probability is improved, it is improved by at least* $\min_{i,j,m} \left|a_i q_m(i) - a_j q_m(j)\right|$, *i.e.,*

$$\epsilon \geq \min_{i,j,m} \left|a_i q_m(i) - a_j q_m(j)\right|. \tag{33}$$

Proof. The difference of the hit probability after and before the local update is

$$\begin{aligned}
[1 - f^{(m)}(\tilde{\mathbf{b}}^{(m)}, \mathbf{b}^{(-m)})] - [1 - f^{(m)}(\mathbf{b}^{(m)}, \mathbf{b}^{(-m)})] &= \sum_{j=1}^{J} a_j \left(\tilde{b}_j^{(m)} - b_j^{(m)}\right) \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}) \\
&= \sum_{j=1}^{J} a_j \left(\tilde{b}_j^{(m)} - b_j^{(m)}\right) q_m(j), \\
&= a_i q_m(i) - a_j q_m(j),
\end{aligned} \tag{34}$$

where we recall that

$$q_m(j) = \sum_{\substack{s \in \Theta \\ m \in s}} p_s \prod_{\ell \in s \setminus \{m\}} (1 - b_j^{(\ell)}),$$

and where we have denoted by $\mathbf{b}^{(-m)}$ the placement strategies of all caches except m. Now, the minimal improvement is given by minimizing (34) over i, j and m. More precisely,

$$\epsilon = \min_{i,j,m} \left|a_i q_m(i) - a_j q_m(j)\right|. \tag{35}$$

The result now follows from $a_i q_m(i) - a_j q_m(j) \geq |a_i q_m(i) - a_j q_m(j)|$. □

Our next result provides more insight into the behaviour of ϵ. The result is expressed in terms of a geometric property of the coverage regions. First, observe that $q_m(j)$ is equal to the sum of the probabilities of being in a partial area (where cache m is included) where file j is missing. To conclude, the function $q_m(j)$ will give a probability value and it consists of a sum of p_s, where $m \in s$ and $s \in \Theta$, values. For any network topology configuration, the p_s will only take a finite number of values. Under *discrete placement* of caches, the minimum difference between two such (non-equal) values is just a function of d and r, which we denote by $\kappa_3(d, r)$ and it scales linearly with the total coverage area (so with the total number of caches N). Hence, we have

$$p_s \neq p_{s'} \Longrightarrow |p_s - p_{s'}| \geq \kappa_3(d, r) N^{-1}. \tag{36}$$

From the polynomial scaling of the popularity distribution we have

$$a_i \geq \kappa_4 J^{-\kappa_2}, \tag{37}$$

for any $1 \leq i \leq J$, with constants $\kappa_2 \geq 0$ and $\kappa_4 > 0$

Lemma 4. *The minimum improvement in hit probability is lower bounded by*

$$\epsilon \geq \kappa_1 N^{-1} J^{-\kappa_2}.$$

Proof. The minimum improvement in hit probability depends both on partial areas and the popularities of the files of interest, as given by Lemma 3. The result now follows from (36) and (37), with $\kappa_1 = \kappa_3(d, r)\kappa_4$. □