Optimization of Caching Devices with Geometric Constraints

Konstantin Avrachenkov, Xinwei Bai, Jasper Goseling

Optimization of Caching Devices with Geometric Constraints

Konstantin Avrachenkov[a], Xinwei Bai[b], Jasper Goseling[b]

[a]*INRIA Sophia Antipolis, France*
[b]*Stochastic Operations Research, University of Twente, The Netherlands*

Abstract

It has been recently advocated that in large communication systems it is beneficial both for the users and for the network as a whole to store content closer to users. One particular implementation of such an approach is to co-locate caches with wireless base stations. In this paper we study geographically distributed caching of a fixed collection of files. We model cache placement with the help of stochastic geometry and optimize the allocation of storage capacity among files in order to minimize the cache miss probability. We consider both per cache capacity constraints as well as an average capacity constraint over all caches. The case of per cache capacity constraints can be efficiently solved using dynamic programming, whereas the case of the average constraint leads to a convex optimization problem. We demonstrate that the average constraint leads to significantly smaller cache miss probability. Finally, we suggest a simple LRU-based policy for geographically distributed caching and show that its performance is close to the optimal.

Keywords: Caching, wireless networks, stochastic geometry

1. Introduction

We consider caching of a collection of files by a set of geographically distributed storage devices with wireless communications capabilities and random network coding. Clients can retrieve cached data from all devices that are within its connectivity radius. Since the caching devices have limited

Email addresses: k.avrachenkov@sophia.inria.fr (Konstantin Avrachenkov), x.bai@utwente.nl (Xinwei Bai), j.goseling@utwente.nl (Jasper Goseling)

storage capacity, not all files can be stored in all caches. Therefore, there is a positive probability that a file that is requested by a client cannot be retrieved from the caching devices that are within range and a cache miss occurs. The general aim of this paper is to optimize the cache allocation so to minimize the cache miss probability.

It has been recently advocated that in large communication systems it is beneficial both for the users and for the network as a whole to store content closer to users. This idea can be realized by Information Centric Networking (ICN), a new paradigm for the network architecture where the data is addressed by its name or content directly rather than by its physical location. There is no predefined location for the data in ICN and the content is naturally cached along the retrieval path. Examples of the ICN architecture are CCN/NDN [1], DONA [2] and TRIAD [3]. Our results can be useful for the design of the wireless networks with the ICN architecture in which case cellular base stations also serve as caches. Wireless sensor networks represent another potential application of our results. Sensors have severe limitation on both memory and transmission capability. It might be useful for sensors to have access to some aggregated characteristics in addition to the local ones. In such a case, our results provide optimal distributed allocation of the aggregated characteristics.

Let us elaborate on the problem formulation in further details. Storage (or caching) devices are placed in the plane according to a homogeneous spatial Poisson process. The homogeneous spatial Poisson process is accepted for modelling the location of base stations providing a good compromise between realistic representation of the wireless network and mathematical tractability [4, 5, 6]. For some cases, e.g., for Sydney base station network [7], it has been shown that the spatial homogeneous Poisson process represents very well the distribution of base stations. In other cases, a non-homogeneous Poisson process can be more appropriate for modelling the distribution of base stations. In fact, some results of the present work can be extended to the case of non-homogeneous Poisson process and we discuss such extensions later in the paper. The size of the file catalog is finite and fixed. A client will request one of the files from the catalog at random according to a known file popularity distribution that is the same for all clients. In particular, for numerical illustration purpose we will consider the case that file popularities follow a Zipf distribution. For the sake of tractable performance evaluation analysis, we make a technical assumption that files consist of the same number of chunks of a fixed size. We suggest to use random linear network coding, in which

case linear combination of chunks can be stored in the caching devices. As was shown in [8], the network coding based allocation strategy outperforms a strategy without coding for a wide range of performance measures and any spatial distribution of caches.

Our interest in the current paper is in the case when the caches are reachable only within a fixed distance to the client. This is a standard model in wireless networks which gives high level but still quite accurate representation of a wireless connection [4, 5]. Our goal is to minimize the cache miss probability, which is the probability that a client cannot get the requested file from the caches within range. Since the probability of not recovering a file from coded chunks is negligible in comparison to the overall cache miss probability, we concentrate solely on the calculation of the cache miss probability and on the optimization of the system with respect to this metric.

We have multiple files and a limited memory in each storage device. Thus, the question is how many linear combinations of each file to store in a particular storage device. Initially, we consider the case when we make the same allocation in all caches, *i.e.*, each cache stores the same number of linear combinations of each file. As a consequence we guarantee a capacity constraint on each individual cache. We formulate an optimization problem with a non-convex objective function and linear constraints. We demonstrate that this problem is a generalization of an unbounded knapsack problem [9]. In particular it is a separable nonlinear integer program, which can be solved using dynamic programming. In addition to providing a formal statement of this result, we give exact closed form results for some special cases of the problem as well as insight into the structure of the solution in the general case.

The above formulation leads to the same allocation in each storage device, which likely leads to inefficient memory utilization and to the lack of file diversity. Thus, we then turn our attention to a relaxation of the problem in which, instead of imposing a hard capacity constraint on each of the caches, we require that the average storage space used in the caches is upper bounded. In particular, we consider cache allocation strategies in which the number of linear combination to store for a file in a cache is a random variable. The number of such combinations is independently and identically decided for each cache. We impose an average capacity constraint on the number of chunks stored in a caching device, where the average is over the caching devices. We analyze the resulting optimal strategy for the case when files

consist of a single piece and show that the performance under an average capacity significantly outperforms the optimal performance under a per cache capacity constraint.

Finally, we consider a dynamic scenario when the clients arrive over time. We study two LRU-based caching policies, cooperative and fully distributed. Both policies demonstrate that performance is not far from the optimal one and that there is a small loss of efficiency in the fully distributed case compared to the cooperative case. This indicates that a simple distributed LRU-based caching policy can be safely deployed in practice for geographically distributed caches. Also, it indicates that our results on the optimal placement policies can provide insight into the performance in the dynamic setting.

Let us outline the organization of the paper. In Section 3 we define the model, discuss the constraints and optimization criterion. The problem with per cache constraints is analysed in Section 4. In particular, we provide structural insight into the optimal storage allocation strategy and show that the problem is a generalization of the unbounded knapsack problem and can be solved by dynamic programming approach. Then, in Section 5 we introduce the average constraint, which makes memory usage more efficient and increases file diversity. In an important particular case we are able to solve the average constraint problem in a closed form. In Section 6 we present distributed and cooperative LRU-based policies. In Section 7 we demonstrate that the performance of the distributed LRU-based policy is not far from the optimal performance. The numerical results of Section 7 also confirm that the average constraint in comparison with the per cache constraint, brings improved efficiency and file diversity. Finally, in Section 8 we provide a discussion of our result and an outlook on future research.

2. Related work

Literature on caching is vast. Therefore, we limit our discussion to work on caching which we feel is most relevant to the present work. The application of network coding for distributed storage is studied in [10] and in [11, 12] specifically for the case of content distribution in wireless networks. The use of coding was also explored in [13] where it was shown how to efficiently allocate the data at caches with the aim of ensuring that any sufficiently large subset of caches can provide the complete data. The difference with the current work is that we are taking the geometry of the deployment of the storage devices into account. In [14], see also [15], coding strategies

for networks of caches are presented, where each user has access to a single cache and a direct link to the source. It is demonstrated how coding helps to reduce the load on the link between the caches and the source. Note that we assume that different transmissions from caches to the clients are orthogonal, for instance by separating them in time or frequency. In [16] the impact of non-orthogonal transmissions is considered and scaling results are derived on the best achievable transmission rates. In [17] a heterogeneous system of small coverage access points and large coverage base stations is considered.

Systems of distributed storage devices or caches can be classified according to the amount of coordination between the devices. In [18] an approach with implicit coordination is proposed. Networks of caches are notoriously difficult to analyze. Only some very particular topologies and caching strategies (see [19] and references therein) or approximations [20, 21] have been studied. In a recent work [22] ergodicity of cache networks has been investigated. Using continuous geometrical constraints on cache placement instead of combinatorial constraints allows us to obtain exact analytical results.

Other work on caching in wireless networks is, for instance, [23, 24, 25]. In [23] the authors analyze the trade-off between energy consumption and the retrieval delay of data from the caches. In [24], the authors consider the optimal number of replicas of data such that the distance between a requesting node and the nearest replica is minimized. Data sharing among multiple caches such that the bandwidth consumption and the data retrieval delay are minimal is considered in [25]. None of [23, 24, 25] are considering coded caching strategies.

We would like to emphasize that except for [11, 12] none of the above mentioned works considered continuous geometric constraints on storage device placement.

Networks of wireless caches in the plane, *i.e.*, with geometric constraints, were first studied in [8] for the case of single file. The tradeoff between the retrieval performance and the deployment cost in terms of number of caches and their capacity was studied in [26]. Both papers considered the storage of just one single data file. In [27, 28] the framework of stochastic geometry was applied to performance evaluation of a network of small base stations with emphasis on physical layer. The question of optimal storage allocation was not studied there. In the recent work [29] a model very similar to our average capacity constraint model has been analysed from a different angle. We would like to note that the present work has been done before [29].

3. Model and Notation

Caching devices are placed in the plane $\mathbb{R}^2$ according to a homogeneous spatial Poisson process with density λ. The spatial Poisson process is known to be an appropriate generic model for location of base stations or sensors in wireless networks [4, 5, 6, 7]. The devices serve as caches for a catalog of L data files. Without loss of generality, we consider a single client that is located at an arbitrary location in the plane [4, 5] and can access only caches within radius r. Since the caches are distributed according to a homogeneous spatial Poisson process, the number of caches within radius r follows a Poisson distribution with parameter $x = \lambda\pi r^2$. That is,

$$P(n \text{ caches within radius } r) = \frac{x^n}{n!}e^{-x}.$$

The parameter x has an interpretation as an expected number of devices inside the area of size πr^2. The client is interested in retrieving one of the L files. The file that is required by the client is selected at random. The probability that the i-th file is selected is p_i, $i = 1, \ldots, L$. Without loss of generality, we assume that $p_1 \geq p_2 \geq \ldots \geq p_L$. The probability distribution p_i represents the popularity of the files. Most of the results in this paper will be obtained for an arbitrary file popularity distribution. In some cases, in particular for illustration of our results by numerical examples, we consider the Zipf distribution. Let p_i^z denote the probability of file i under a Zipf distribution [30] with parameter $s > 0$, *i.e.*, $p_i^z = i^{-s}/\sum_{k=1}^{L} k^{-s}$.

We suppose that files consist of N chunks (ICN terminology). For the sake of tractability, we assume that all packets of all files are of the same size. Each caching device can store at most C packets. We assume that $C < LN$, *i.e.*, we cannot store all files in a device. Therefore, a means of allocating (parts of) files to caches needs to be devised. Inevitably, for any allocation strategy there will be a positive probability that the client cannot retrieve the desired file from the caches within its range. Our interest in this paper is in minimizing this probability, the cache miss probability, by optimizing the storage strategy. We allow for caches to store only part of a file. Also, we allow for random linear network coding to be used. As a consequence, caches do not store the data packets themselves, but store instead one or more random linear combinations of the data packets of a file. The purpose of using network coding is that with high probability in order to recover a file it is sufficient for the client to retrieve any N linear combinations of packets.

In a network of caches the probability that the client cannot recover a file from N linearly coded packets is negligible compared to the overall cache miss probability [8, 31]. Therefore, we ignore this event in this paper and use the following assumption.

Assumption 1. *A file can be recovered from any set of N linear combinations of packets from that file.*

The storage strategy is based on storing in each caching device n_i linear combinations of the packets of file i. Since the considered model is space homogeneous, all caching devices follow the same caching strategy. The capacity constraint that we need to satisfy is

$$\sum_{i=1}^{L} n_i \leq C. \tag{1}$$

Now in order to retrieve file i the client needs to obtain at least N linear combinations for that file. If the caches within radius r cannot provide these linear combinations, a cache miss occurs. Our interest in this paper is in minimizing the cache miss probility by optimizing the values n_i, $i = 1, \ldots, L$. The probability is over the placement of the caches as well as the selection of the file by the client. More precisely, our performance measure of interest is

$$P_{\mathrm{e}} = \sum_{i=1}^{L} p_i P_{\mathrm{e}}(i), \tag{2}$$

where

$$P_{\mathrm{e}}(i) = \Pr\{\text{client cannot obtain file } i \text{ from caches within distance } r\}. \tag{3}$$

In this section we have defined only the problem with per cache capacity constraints. The relaxation to average constraints is defined and analysed in Section 5. In the next section we first analyze the case of per cache capacity constraints. Then, in Section 6 we consider a dynamic scenario with arriving and departing users.

4. Individual Cache Capacity Constraints

We start this section with a formulation of the optimization problem in Subsection 4.1. Next we provide some results on the structure of the optimal

solution to this problem in Subsection 4.2. In Subsection 4.3 we give an analytical expression for the optimal solution for the case that files consist of a single chunk. Finally, in Subsection 4.4 we provide a dynamic programming approach for solving the general case.

4.1. Formulation of optimization problem

The client can connect to all caches that are within radius r. Since the caches are distributed according to a homogeneous Poisson process, the number of caches within radius r follows a Poisson distribution with parameter $x = \lambda\pi r^2$. When the client wants file i, $\lceil N/n_i \rceil$ caches are needed to get the complete file. This request will be missed if there are less than $\lceil N/n_i \rceil$ caches within radius r to the client. Therefore, the miss probability for file i is given by

$$\begin{aligned} P_{\mathrm{e}}(i) &= \sum_{k=0}^{\lceil N/n_i \rceil - 1} P(k \text{ caches within radius } r) \\ &= \sum_{k=0}^{\lceil N/n_i \rceil - 1} \frac{x^k}{k!} e^{-x} \\ &= Q(\lceil N/n_i \rceil, x), \end{aligned} \tag{4}$$

where Q is the regularized incomplete Gamma function. Since file i is requested with probability p_i, the expected miss probability is

$$P_{\mathrm{e}} = \sum_{i=1}^{L} p_i Q(\lceil N/n_i \rceil, x).$$

For notational convenience, let the function f be defined as

$$f(n_i) = Q(\lceil N/n_i \rceil, x). \tag{5}$$

From the above it follows that the minimization of the cache miss probability P_e is given by the following optimization problem.

Problem 1.

$$\begin{aligned} \min \quad & \sum_{i=1}^{L} p_i f(n_i) \\ \text{subject to} \quad & \sum_{i=1}^{L} n_i = C, \\ & n_i \in \mathbb{N},\ i = 1, \ldots, L. \end{aligned} \tag{6}$$

Note that the objective function of this optimization problem is non-increasing in n_i. However, for $N > 1$ it is not convex. Since the objective function is non-increasing, we consider only equality $\sum_{i=1}^{L} n_i = C$ in the capacity constraint.

4.2. Structure of the optimal solution

Our first result deals with the structure of the optimal solution. In particular we demonstrate that the number of linear combinations stored for file i is a non-increasing function in i.

Theorem 1. *Let $\bar{n} = (\bar{n}_1, \ldots, \bar{n}_L)$ be an optimal solution to Problem 1. Then $\bar{n}_1 \geq \bar{n}_2 \geq \cdots \geq \bar{n}_L$.*

Proof. Suppose there exists $j < k$ for which $n_j < n_k$, then consider n', constructed by having $n'_j = n_k$, $n'_k = n_j$ and the others remain the same, then we can get

$$\begin{aligned}
&\sum_{i=1}^{L} p_i f(n'_i) - \sum_{i=1}^{L} p_i f(n_i) \\
&\qquad = p_j f(n'_j) + p_k f(n'_k) - p_j f(n_j) - p_k f(n_k) \\
&\qquad = p_j f(n_k) + p_k f(n_j) - p_j f(n_j) - p_k f(n_k) \\
&\qquad = (p_j - p_k)[f(n_k) - f(n_j)].
\end{aligned}$$

Since $j < k$, then $p_j - p_k \geq 0$. Also, since f is non-increasing in n_i and $n_j < n_k$, then we can get that $f(n_j) \geq f(n_k)$, *i.e.*, $f(n_k) - f(n_j) \leq 0$. Therefore, $f(n') - f(n) \leq 0$, which means that after the exchange, the objective value will not become higher. Then we can keep doing exchange until $n_1 \geq n_2 \geq \ldots \geq n_L$. □

As we can see, if a per cache constraint is used, the optimal allocation is the same for all caches. This will likely result in inefficient memory usage and the total absence of some files from the caching system. In Section 5 we introduce the average capacity constraint which will help to mitigate these issues.

4.3. Optimal solution for $N = 1$

Next, we consider the case that files consist of a single chunk, *i.e.*, $N = 1$. This implies that we either store a file completely in each cache, or not at

all, *i.e.*, n_i can be either 0 or 1. If $n_i = 1$, then file i is stored in every cache. In this case when a client requests file i, the miss probability will be

$$P_{\mathrm{e}}(i) = e^{-x}.$$

If $n_i = 0$, the miss probability will be 1 if it is requested. Therefore, we can see that

$$P_{\mathrm{e}}(i) = \begin{cases} e^{-x}, & \text{if } n_i = 1, \\ 1\ , & \text{if } n_i = 0 \end{cases} \tag{7}$$

and we can write equation (7) as

$$P_{\mathrm{e}}(i) = e^{-n_i x}. \tag{8}$$

For the special case of $N = 1$, the general optimization problem, Problem 1, reduces to the following problem.

Problem 2.

$$\begin{aligned} \min \quad & \sum_{i=1}^{L} p_i e^{-n_i x} \\ \text{subject to} \quad & \sum_{i=1}^{L} n_i = C \\ & n_i \in \{0,1\},\ i = 1, \ldots, L. \end{aligned}$$

Since n_i is binary and we know that the optimal solution has a structure $n_1 \geq n_2 \geq ...n_L$, it follows directly from Theorem 1 that the optimal solution of Problem 2 is as stated in the following result.

Corollary 1. *The optimal solution of Problem 2 is* $\bar{n} = (\bar{n}_1, \bar{n}_2, ..., \bar{n}_L)$*, where*

$$\bar{n}_i = \begin{cases} 1, \text{ if } i \leq C, \\ 0, \text{ if } i > C. \end{cases}$$

Note that contrary to the case $N > 1$ the objective function of Problem 2 is convex. We will make use of this property in Section 5, where we will revisit the case $N = 1$ under an average capacity constraint.

4.4. Dynamic Programming

In this section we return to the general case of arbitrary N. As already discussed in Section 1, Problem 1 is a generalization of the unbounded knapsack problem. The generalization comes from the fact that the objective function is not a weighted sum of the variables n_i, but a non-convex function in these variables. It is well-known that the unbounded knapsack problem can be solved in pseudo-polynomial time using dynamic programming [9]. In this section we demonstrate that Problem 1 can also be solved using dynamic programming.

In order to formulate a dynamic programming solution we interpret Problem 1 as follows. We have C units in total, and there are L slots to put the units in. Assigning n_i units to slot i induces a certain cost. Our goal is to distribute all of the C units over these slots with a minimal total cost, which is defined as $\sum_{i=1}^{L} p_i f(n_i)$, where f is defined in (5). The idea of dynamic programming is to assign the units one by one, leading to a recursive procedure in both L and C.

More precisely, consider the problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{\ell} p_i f(n_i), \\ \text{subject to} \quad & \sum_{i=1}^{\ell} n_i = c, \end{aligned} \tag{9}$$

and let $F(\ell, c)$ denote its optimal value. Our interest is in $F(L, C)$ and $\bar{n}$, a solution attaining $F(L, C)$. For $\ell = 2, \ldots, L$ and $c = 0, \ldots, C$ let

$$\tilde{n}_{\ell,c} = \operatorname*{argmin}_{n \in \{0,\ldots,c \wedge N\}} \left\{ F(\ell - 1, c - n) + p_\ell f(n) \right\}, \tag{10}$$

where $c \wedge N = \min\{c, N\}$. The dynamic programming approach to Problem 1 is based on the observation that for $2 \leq \ell \leq L$ and $0 \leq c \leq C$ we can express $F(\ell, c)$ as

$$F(\ell, c) = F(\ell - 1, c - \tilde{n}_{\ell,c}) + p_\ell f(\tilde{n}_{\ell,c}). \tag{11}$$

The procedure is initialized by considering $\ell = 1$ and $0 \leq c \leq C$, for which we know that the optimal value $F(1, c) = p_1 f(c \wedge N)$. Next we apply formula (11) iteratively. After computing all values $F(\ell, c)$, the solution $\bar{n}$ can be constructed from the values of $\tilde{n}_{\ell,c}$ by tracking backwards starting at

Algorithm 1 Dynamic Programming Algorithm for Problem 1

for $c = 0 : C$ **do**

$$F(1,c) = \begin{cases} p_1 f(c), & \text{if } c \leq N, \\ p_1 f(N), & \text{if } c > N. \end{cases}$$

end for

for $\ell = 2 : L$ **do**

for $c = 0 : C$ **do**

$\tilde{n}_{\ell,c} = \operatorname{argmin}_{n \in \{0,\ldots,N \wedge c\}} \{F(\ell - 1, c - n) + p_\ell f(n)\}$,

$F(\ell, c) = F(\ell - 1, c - \tilde{n}_{\ell,c}) + p_\ell f(\tilde{n}_{\ell,c})$.

end for

end for

$c = C$

for $\ell = L : -1 : 2$ **do**

$\bar{n}_\ell = \tilde{n}_{\ell,c}$,

$c = c - \bar{n}_\ell$.

end for

$\bar{n}_1 = c$,

$P_e = F(L, C)$.

$\ell = L$. The complete procedure is presented as Algorithm 1. The following theorem provides a formal statement of the result. The proof follows from standard results on dynamic programming.

Theorem 2. *Algorithm 1 provides a globally optimal solution to Problem 1 in pseudo-polynomial time.*

In Section 7 we will provide additional insight into the optimal solution of Problem 1.

5. Average Capacity Constraints

In this section, instead of imposing an individual per cache constraint on each of the devices, we require that the average storage space used in the devices is upper bounded. We analyze the resulting optimal strategy for the case that files consist of a single chunk ($N = 1$) and show that the performance under an average capacity constraint significantly outperforms the optimal performance under a per cache capacity constraint.

Since files consist of a single chunk, the choice to make is whether to store the complete file or not to store the file at all. The proposed strategy places file i in a cache with probability q_i. Placement of files is independent between caches. By the independence of the placement over the caches and the thinning property of the Poisson process [4], it follows that those caches that contain file i are again distributed according to a spatial Poisson process, this time with density $q_i\lambda$. Therefore, the probability that file i cannot be retrieved from the caches within distance r is

$$P_{\mathrm{e}}(i) = p_i e^{-q_i x}, \tag{12}$$

with $x = \lambda\pi r^2$.

Now the goal is to optimize $\sum_{i=1}^{L} P_{\mathrm{e}}(i)$ subject to the capacity constraint. This leads to the following optimization problem.

Problem 3.

$$\begin{aligned} \min \quad & \sum_{i=1}^{L} p_i e^{-q_i x} \\ \text{subject to} \quad & \sum_{i=1}^{L} q_i = C, \\ & 0 \le q_i \le 1, \ i = 1, \ldots, L. \end{aligned}$$

Note that contrary to the objective function of Problem 1, the above objective function is convex. Also note that in contrast to Problem 2 the variables in Problem 3 are continuous. Therefore, Problem 3 is a convex optimization problem.

5.1. Optimal solution

Since Problem 3 is convex, the Karush-Kuhn-Tucker (KKT) conditions provide necessary and sufficient conditions for optimality. We will construct an explicit analytical solution to Problem 3 that satisfies the KKT conditions.

The Lagrangian function corresponding to Problem 3 is

$$\begin{aligned} L(q, \nu, \lambda, \omega) = \sum_{i=1}^{L} p_i e^{-q_i x} + \nu\Big(\sum_{i=1}^{L} q_i - C\Big) \\ - \sum_{i=1}^{L} \lambda_i q_i + \sum_{i=1}^{L} \omega_i (q_i - 1), \end{aligned} \tag{13}$$

where $q, \lambda, \omega \in \mathbb{R}^L_+, \nu \in \mathbb{R}$.

Let $\bar{q}, \bar{\lambda}, \bar{\omega}$ and $\bar{\nu}$ be primal and dual optimal. Then the KKT conditions for Problem 3 state that

$$\begin{aligned}
0 \leq \bar{q}_i &\leq 1, && (14)\\
\sum_{i=1}^{L} \bar{q}_i &= C, && (15)\\
\bar{\lambda}_i &\geq 0, \ \forall i = 1, ..., L, && (16)\\
\bar{\omega}_i &\geq 0, \ \forall i = 1, ..., L, && (17)\\
\bar{\lambda}_i \bar{q}_i &= 0, \ \forall i = 1, ..., L, && (18)\\
\bar{\omega}_i(\bar{q}_i - 1) &= 0, \ \forall i = 1, ..., L, && (19)\\
-p_i x e^{-\bar{q}_i x} + \bar{\nu} - \bar{\lambda}_i + \bar{\omega}_i &= 0, \ \forall i = 1, ..., L. && (20)
\end{aligned}$$

For notational convenience we introduce the functions $g_i : \mathbb{R} \to [0, 1]$, $i = 1, \ldots, L$, as follows

$$g_i(\nu) = \begin{cases} 1, & \text{if } \nu \leq p_i x e^{-x}, \\ \dfrac{1}{x} \log \dfrac{p_i x}{\nu}, & \text{if } p_i x e^{-x} < \nu < p_i x, \\ 0, & \text{if } \nu \geq p_i x. \end{cases} \quad (21)$$

Furthermore, let $g : \mathbb{R} \to [0, L]$ be defined as $g(\nu) = \sum_{i=1}^{L} g_i(\nu)$. Observe that $g(\nu) = L$ for $\nu \in (-\infty, p_L x e^{-x}]$, that $g(\nu) = 0$ for $\nu \in [p_1 x, \infty)$ and that it is strictly decreasing in the interval $(p_L x e^{-x}, p_1 x)$.

Lemma 1. *Let $\bar{q}$ and $\bar{\nu}$ be optimal. Then $\bar{q} = (g_1(\bar{\nu}), \ldots, g_L(\bar{\nu}))$.*

Proof. Let $i \in \{1, \ldots, L\}$. From (18), (19) and (20), we have

$$\bar{\omega}_i = \bar{q}_i(p_i x e^{-\bar{q}_i x} - \bar{\nu}), \quad (22)$$

which, when inserted into (19), gives

$$\bar{q}_i(\bar{q}_i - 1)(p_i x e^{-\bar{q}_i x} - \bar{\nu}) = 0. \quad (23)$$

From (23), we see that $0 < \bar{q}_i < 1$ only if $\bar{\nu} = p_i x e^{-\bar{q}_i x}$. Since $0 \leq q_i \leq 1$, this implies that $\nu \in [p_i x e^{-x}, p_i x]$.

If $\bar{\nu} < p_i x e^{-x}$, then

$$\bar{\omega}_i = \bar{\lambda}_i + p_i x e^{-\bar{q}_i x} - \bar{\nu} > 0.$$

Thus, from (19), we have $q_i = 1$. Similarly, if $\bar{\nu} > p_i x$, it follows from

$$\bar{\lambda_i} = \bar{\omega_i} + \bar{\nu} - p_i x e^{-\bar{q}_i x} > 0$$

and (18) that $\bar{q}_i = 0$. □

It remains to solve for $\bar{\nu}$. The complete solution is provided by the next theorem.

Theorem 3. *The optimal solution of Problem 3 is given by*

$$\bar{q}_i = \begin{cases} 1 \quad , & \text{if } i < k_1, \\ \dfrac{1}{x} \log \dfrac{p_i x}{\bar{\nu}}, & \text{if } k_1 \leq i \leq k_2, \\ 0 \quad , & \text{if } i > k_2 \end{cases}$$

where k_1, k_2 are given by

$$k_1 = \min \left\{ 1 \leq \ell \leq L \middle| \; g(p_\ell x e^{-x}) \geq C \right\}, \tag{24}$$

$$k_2 = \max \left\{ 1 \leq \ell \leq L \middle| \; g(p_\ell x) \leq C \right\} \tag{25}$$

and

$$\bar{\nu} = \exp \left\{ \frac{1}{k_2 - k_1 + 1} \sum_{j=k_1}^{k_2} \log p_j x - \frac{x(C - k_1 + 1)}{k_2 - k_1 + 1} \right\}. \tag{26}$$

Proof. From Lemma 1 it follows that there exist $k_1, k_2 \in [1, L]$ such that $\bar{q}_1 = \bar{q}_2 = \cdots = \bar{q}_{k_1 - 1} = 1$ and $\bar{q}_{k_2+1} = \bar{q}_{k_2+2} = \cdots = \bar{q}_L = 0$. In particular, k_1 is given by

$$k_1 = \min \left\{ 1 \leq \ell \leq L \middle| \; \bar{v} > p_\ell x e^{-x} \right\}. \tag{27}$$

Note that the above minimum is guaranteed to exist, because otherwise $q_i = 1$ for all $i = 1, \ldots, L$, leading to a contradiction on the assumption that $\sum_{i=1}^{L} q_i = C < L$. Condition (24) is obtained by applying the non-increasing function g to the LHS and the RHS in the constraint in (27) and by observing that from Lemma 1 and (15) it follows that $g(\bar{\nu}) = C$.

Condition (25) follows in similar lines by starting from

$$k_2 = \max \left\{ 1 \leq \ell \leq L \middle| \; \bar{v} < p_\ell x \right\}. \tag{28}$$

This maximum exists, because otherwise $\sum_{i=1}^{L} q_i = 0$, which contradicts (15). Finally, the proof of the lemma is completed by solving for $\bar{\nu}$ in $g(\bar{\nu}) = C$. □

We note that in contrast to the solution of Problem 1 the solution in the case of the average capacity constraint admits a probabilistic policy of file placement. This should improve the system efficiency as well as file diversity. We will further illustrate the results of Theorem 3 in Section 7.

6. Dynamic Setting

In this section we consider a dynamic scenario in which clients arrive over time. More precisely, we consider $L > 1$ files, each of $N = 1$ packets. Clients arrive over time. We assume that at any time there is at most one client, *i.e.*, we assume that the request of a user is completely handled and that the caches are updated before the next client arrives. This assumption is just for simplicity of modelling and could be safely neglected in a real implementation. The files that are requested by users are selected at random according to a Zipf distribution, independently across users. Clients arrive to random locations in the plane. As in the other parts of this paper, a client can connect to all caches that are within range r of the client.

The caching policy is as follows. If the file requested by the user is present in any of the caches that are within its range, the file is delivered to the client from the cache. If the requested file is not present in the cache it is fetched from the server and delivered to the user. In addition, the file is then placed in the cache that is closest to the user.

Each cache individually follows the Least Recently Used (LRU) policy for caching files. This means that each cache keeps an ordered list of the files that are locally cached. If a file is served to a client it is moved to the head of the list. If a file was fetched from the server it will be placed at the head of the list in the cache that is closest to the user that is requesting that file. If the number of files in the list is exceeding the cache capacity the file at the tail of the list is dropped from the cache.

Note that in the caching policy as described above there is no cooperation between caches. It is a straightforward extension of the LRU policy to a network of caches. In particular, the caching policy does not make use of information about the file popularity. In this section we are interested in comparing the performance of this very simple policy with the optimal allocation strategy of Section 5. In order to do so we have simulated the LRU policy as described above. The numerical results are presented in Section 7.

In addition to the comparison with the optimal allocation strategy we consider the performance of an LRU policy in which the caches fully cooperate. More precisely, we analyze the performance of a single LRU cache with a capacity that equals the expected sum capacity of all caches that are within range of a client. This allows us to evaluate the 'penalty to pay' for distributing cache capacity over several caches that operate independently. The expected number of caches that are within range of a client is $\lambda \pi r^2$.

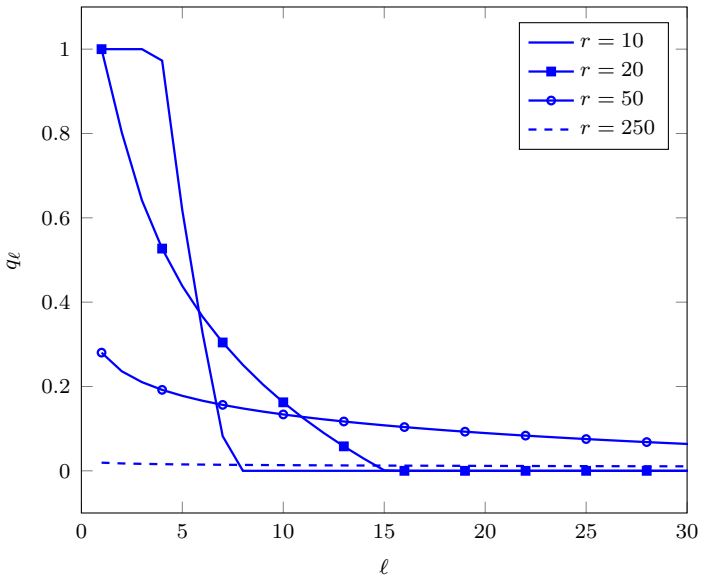

Figure 1: Optimal allocation probabilities under an average capacity constraint. ($L = 2000$, $N = 1$, $C = 5$, $\lambda = 2 \cdot 10^{-3}$, $s = 1$)

The expected sum capacity is therefore $C\lambda\pi r^2$. The cache miss probability of a single LRU cache is known to be accurately approximated with the Che approximation [32, 33]. We provide the numerical evaluation of this approximation for a cache of capacity $C\lambda\pi r^2$ in Section 7.

7. Numerical Evaluation

In this section we present a numerical evalution of the results obtained in the previous sections of this paper. In particular, we consider the case of file popularities following a Zipf distribution, *i.e.*, $p_i^z = i^{-s}/\sum_{k=1}^{L} k^{-s}$, with parameter s. The numerical illustrations will provide some additional insights into the behavior of the optimal cache allocation policies as well as into the behavior of the proposed LRU strategies. In particular, we compare cooperative and fully distributed LRU-based caching policies.

7.1. The optimal solution under an average capacity constraint

Theorem 3 provides an analytical expression for the optimal allocation probabilities under an average capacity constraint. It is not immediately clear from Theorem 3 how k_1 and k_2 depend on, for instance, $x = \lambda\pi r^2$.

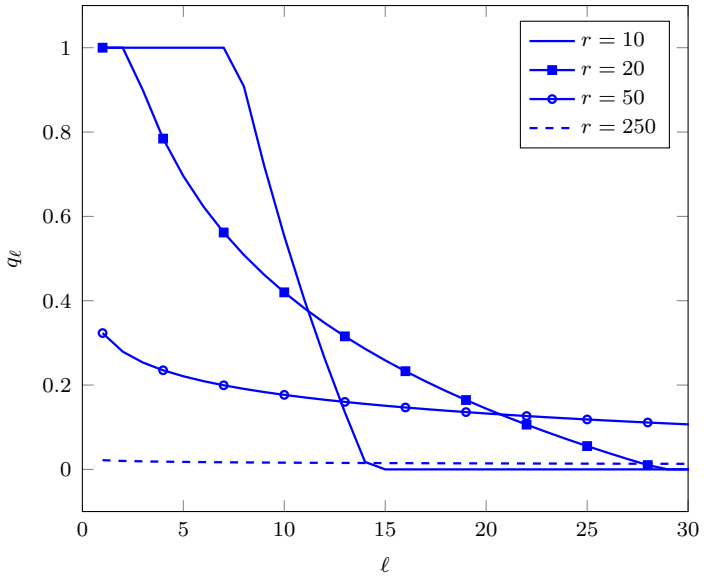

Figure 2: Optimal allocation probabilities under an average capacity constraint. ($L = 2000$, $N = 1$, $C = 10$, $\lambda = 2 \cdot 10^{-3}$, $s = 1$)

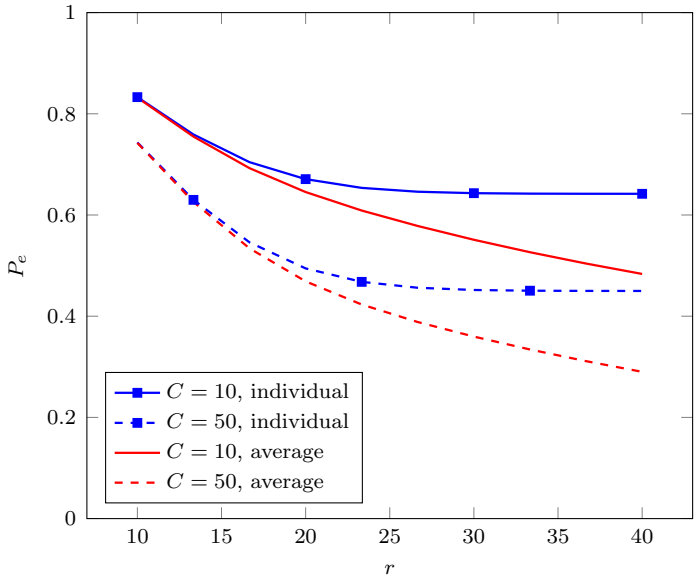

Figure 3: Cache miss probability under individual and average cache capacity constraints. ($L = 2000, N = 1, \lambda = 2 \cdot 10^{-3}, s = 1$)

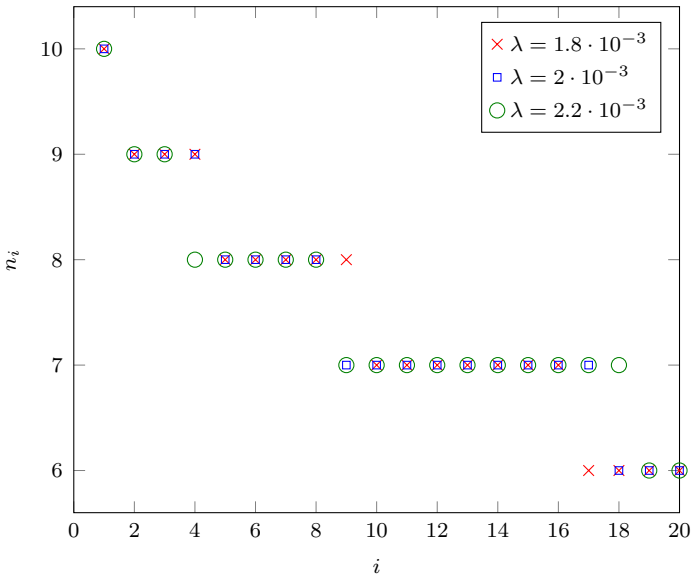

Figure 4: Influence of small differences in λ on storage policy under individual capacity constraints ($L = 20$, $N = 50$, $C = 150$, $s = 1$, $r = 50$)

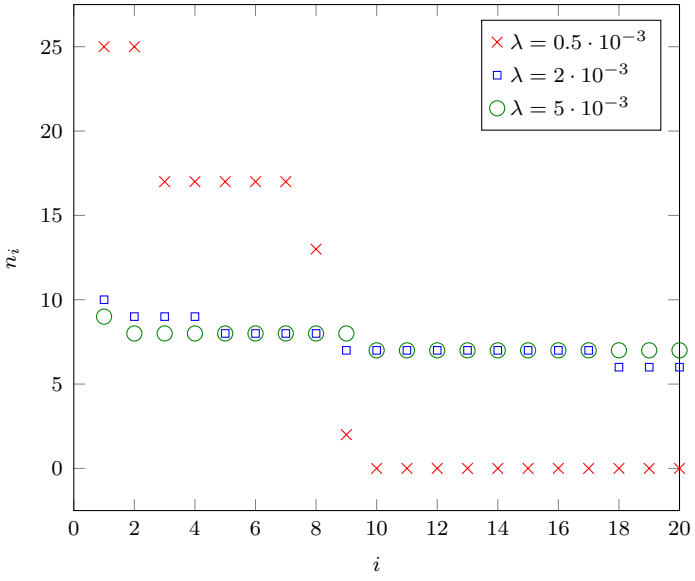

Figure 5: Influence of large differences in λ on storage policy under individual capacity constraints ($L = 20$, $N = 50$, $C = 150$, $s = 1$, $r = 50$)

In Figures 1 and 2 we have illustrated the optimal allocation probabilities $\bar{q}_i$ under an average capacity constraint for various values of r. We observe that if r is large, which means that the client can reach more caches within the range, then we store all files with equal probability C/L. It is intuitively clear that this minimizes the cache miss probability, since now all files can be retrieved with high probability. If r is small and less caches can be reached, we will put priority, *i.e.*, higher $\bar{q}_i$, on the more popular files.

7.2. Performance under individual and average cache capacity constraints

Next we compare the miss probability of the optimal cache allocation under the individual cache capacity constraints with the miss probability under the average constraint. In Figure 3 we have depicted the cache miss probability as a function of r for two values of C. From the discussion it should be clear that in the limit of large r the cache miss probability under an average capacity constraint should approach zero. This is indeed reflected in Figure 3. The individual capacity constraint, in stark contrast, results in a significant cache miss probability even at large r. The reason is that some files will not be stored at all and, therefore, a request for these files will always result in a cache miss.

Another interpretation of the significant improvement that is offered by allowing an average constraint, which means that whether the file is in the cache or not is probabilistic, is that different caches may have different files and that can help improve the performance and file diversity.

7.3. Non-homogeneous distribution of base stations

Here we argue that if the density of base stations λ does not change very rapidly, our analysis remains applicable but of course approximate. Figure 3 gives performance as a function of r. What is important is that $x = \lambda\pi r^2$ is the only factor of influence. Therefore, our Figure 3 already gives some insight in the behavior as a function of λ. The storage policy under hard constraints is not influenced by the value of x if $N = 1$; we simply store the most popular files. For $N > 1$ we evaluate our dynamic programming policy. Figure 4 demonstrates the influence of λ on the storage policy. The figure provides the number of fragments stored n_i for file i for various values of λ. The figure demonstrates that the policy is not changing much by small perturbations of λ. Hence, if the density of base stations does not change much in space one can use a single policy everywhere without much damage to the system performance. If the density of base stations changes

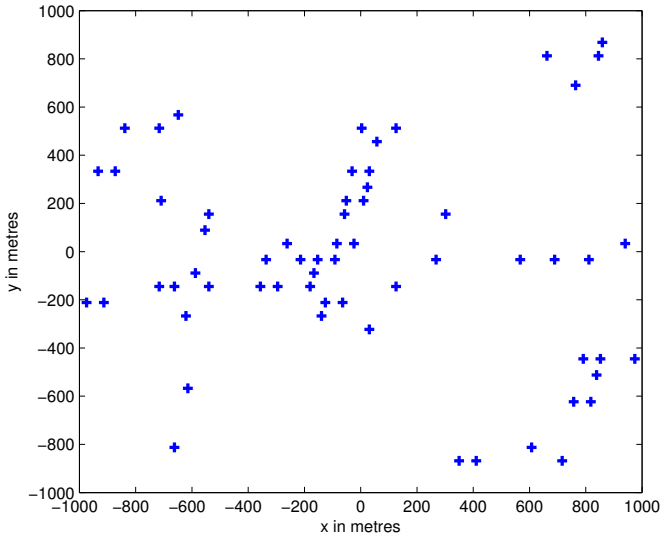

Figure 6: Location of Base Stations from OpenMobileNetwork dataset.

significantly but not too rapidly, as mentioned above, we expect that our results are still practically applicable. Of course, as we demonstrate in the following Figure 5, the optimal policies for different densities of base station distribution can be quite different.

Next we evaluate the performance of coded and uncoded strategies on the topology of a real wireless network. Similar to our study in [8] we have taken the positions of 3G base stations provided by the OpenMobileNetwork project [34]. The base stations are situated in the area 1.95×1.74 kms around the TU-Berlin campus. One can see the positions of the base stations from the OpenMobileNetwork project in Figure 6. We note that the base stations of the real network are more clustered then in a typical realization of a Poisson process, because they are typically situated along roads. We will analyze the performance of our placement strategies (which are optimal for a Poisson network) on this non-Poisson topology. There are 62 base stations in our dataset, corresponding to an average density of $\lambda = 1.8324 \cdot 10^{-5}$. We use this density to derive the optimal placement strategies under individual and average capacity constraints for various values of r. The results are depicted in Figure 7, which also includes the results for a Poisson network

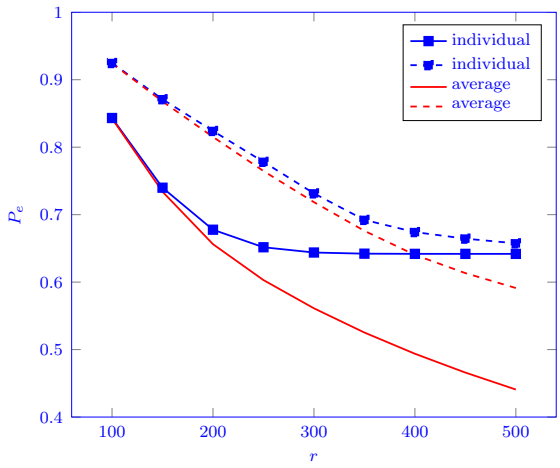

Figure 7: Cache miss probability in Berlin network and Poisson process. In solid line the performance of the Poisson process. In dashed line the performance of the Berlin network. $(C = 10, L = 2000, \lambda = 1.8324 \cdot 10^{-5}, s = 1)$

with the same density. We observe that the clustering increases the cache miss probability, but that our results on the Poisson model approximate the performance on the real data set quite well. The difference between the Poisson case and our dataset is smaller for individual capacity constraints than it is for average capacity constraints. We cannot explain this difference with our current results and suggest as future work to develop an insight into this behavior.

7.4. Dynamic Setting

Finally we consider the dynamic setting of Section 6. In Figures 8 and 9 we have depicted the cache miss probability as a function of the connection range r for cache capacities $C = 10$ and $C = 50$, respectively. In solid lines we have depicted the performance of the fully distributed LRU policy. In dashed lines we have depicted the performance under the optimal allocation strategy of Section 5. Finally, we have depicted in dotted lines the performance of a 'centralized' LRU policy, *i.e.*, we depict the performance of a single LRU cache with capacity equal to expected sum capacity of all caches that are within range of a client.

From Figures 8 and 9 it is clear that the performance of the fully distributed LRU policy is not too far from the performance of the optimal allocation strategy of Section 5. Another observation is that the performance

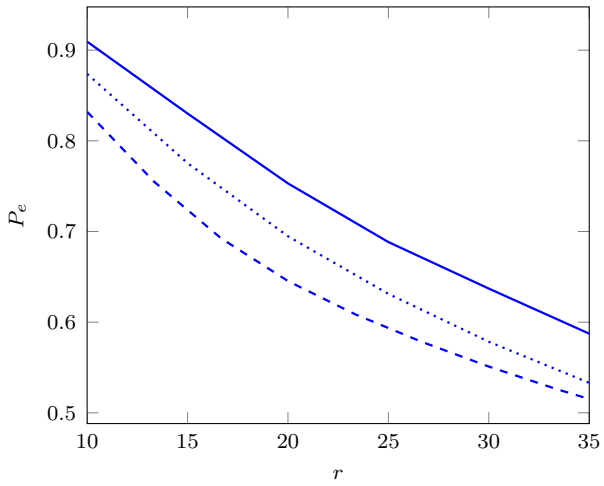

Figure 8: Dynamic scenario. In solid line performance of the fully distributed LRU policy. In dashed line the performance under optimal allocation of Section 5. In dotted line the performance of a single LRU cache with capacity equal to expected sum capacity of all caches that are within range of a client. ($C = 10$, $L = 2000$, $\lambda = 2 \cdot 10^{-3}$, $s = 1$)

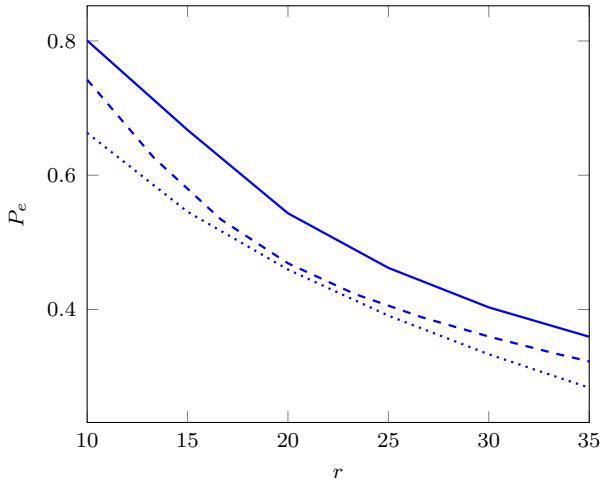

Figure 9: Dynamic scenario. In solid line performance of the fully distributed LRU policy. In dashed line the performance under optimal allocation of Section 5. In dotted line the performance of a single LRU cache with capacity equal to expected sum capacity of all caches that are within range of a client. ($C = 50$, $L = 2000$, $\lambda = 2 \cdot 10^{-3}$, $s = 1$)

difference between our distributed and the 'centralized' LRU policy is small. Therefore, our distributed LRU-based policy with caching in a closest storage device can be safely employed in practice for geographically distributed caching.

8. Discussion

In the current paper we have obtained structural insight into optimal storage allocation strategies in a network of wireless caching devices in a stochastic geometry. We have seen that for the design of geographically distributed caching devices it is better to use average than per cache capacity constraint. We indicate that our model can be practically applied even for non-homogeneous distribution of base stations when the rate of density change is not too rapid. We have also considered a dynamic setting for which we proposed a simple distributed LRU-based policy. We have shown that the performance of this LRU policy is not far from the optimal one, and consequently, this LRU-based policy can be safely employed in practice for geographically distributed caching. Part of the analysis in this paper considered the particular case that files consist of a single packet. In future work we will generalize this analysis. In addition we will consider the dynamic setting in more detail by extending the model to include the latencies of fetching a file from a server and analyzing the overall file delivery latency. In particular, the aim is to obtain a more fundamental insight into the behavior of LRU and others replacement policies in networks of wireless caches in a stochastic geometry setting. A first step in understanding this behavior is to generalize the cache placement strategies from this paper to strategies that allow for a different (deterministic) placement of files in each of the caches. The optimal hit probability under such strategies can then serve as a baseline for online dynamic strategies. Also, it will enable to study non-homogeneous spatial Poisson processes for base station placement or more general placement in a natural way. Investigating different placement in each cache is part of our ongoing efforts as well as [35]. Analyzing various online dynamic strategies can be approached by considering TTL caches [19], which are known to cover many other strategies by carefully choosing the TTL distributions [36].

Acknowledgement

This work was performed while Xinwei Bai was visiting INRIA Sophia Antipolis in fall 2013. This work was supported in part by the Netherlands

Organization for Scientific Research (NWO) grant 612.001.107.

References

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, ser. CoNEXT '09. New York, NY, USA: ACM, 2009, pp. 1–12. [Online]. Available: http://doi.acm.org/10.1145/1658939.1658941

[2] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, ser. SIGCOMM '07. New York, NY, USA: ACM, 2007, pp. 181–192. [Online]. Available: http://doi.acm.org/10.1145/1282380.1282402

[3] M. Gritter and D. R. Cheriton, "An architecture for content routing support in the internet," in *Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems - Volume 3*, ser. USITS'01. Berkeley, CA, USA: USENIX Association, 2001, pp. 4–4. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251440.1251444

[4] F. Baccelli and B. Blaszczyszyn, *Stochastic Geometry and Wireless Networks, Volume I - Theory*, ser. Foundations and Trends in Networking Vol. 3: No 3-4, pp 249-449. NoW Publishers, 2009, vol. 1.

[5] ——, *Stochastic Geometry and Wireless Networks, Volume II-Applications*, ser. Foundations and Trends in Networking Vol. 4: No 1-2, pp 1-312. NoW Publishers, 2009.

[6] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks," *Selected Areas in Communications, IEEE Journal on*, vol. 27, no. 7, pp. 1029–1046, 2009.

[7] C.-H. Lee, C.-Y. Shih, and Y.-S. Chen, "Stochastic geometry based models for modeling cellular networks in urban areas," *Wireless networks*, vol. 19, no. 6, pp. 1063–1072, 2013.

[8] E. Altman, K. Avrachenkov, and J. Goseling, "Distributed storage in the plane," in *IFIP Networking 2014 Conference*, Jun. 2014.

[9] S. Martello and P. Toth, *Knapsack problems*. Wiley New York, 1990.

[10] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.

[11] N. Golrezaei, A. Molisch, A. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *Communications Magazine, IEEE*, vol. 51, no. 4, pp. 142–149, April 2013.

[12] M. Ji, G. Caire, and A. F. Molisch, "The throughput-outage tradeoff of wireless one-hop caching networks," *arXiv preprint arXiv:1312.2637*, 2013.

[13] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes," in *Proceedings of the 4th international symposium on Information Processing in Sensor Networks*. IEEE Press, 2005, p. 15.

[14] M. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. PP, no. 99, pp. 1–1, 2014.

[15] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," in *Communications (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1878–1883.

[16] U. Niesen, D. Shah, and G. W. Wornell, "Caching in wireless networks," *IEEE Trans. Inf. Theory*, vol. 58, no. 10, pp. 6524–6540, 2012.

[17] J. Hachem, N. Karamchandani, and S. Diggavi, "Coded caching for heterogeneous wireless networks with multi-level access," *arXiv preprint arXiv:1404.6560*, 2014.

[18] E. Rosensweig and J. Kurose, "Breadcrumbs: Efficient, best-effort content location in cache networks," in *INFOCOM 2009, IEEE*, 2009, pp. 2631–2635.

[19] 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.

[20] E. Rosensweig, J. Kurose, and D. Towsley, "Approximate models for general cache networks," in *INFOCOM, 2010 Proceedings IEEE*, 2010, pp. 1–9.

[21] H. Che, Y. Tung, and Z. Wang, "Hierarchical web caching systems: modeling, design and experimental results," *IEEE J.Sel. A. Commun.*, vol. 20, no. 7, pp. 1305–1314, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/JSAC.2002.801752

[22] E. Rosensweig, D. Menasche, and J. Kurose, "On the steady-state of cache networks," in *INFOCOM, 2013 Proceedings IEEE*, 2013, pp. 1–9.

[23] P. Nuggehalli, V. Srinivasan, and C.-F. Chiasserini, "Energy-efficient caching strategies in ad hoc wireless networks," in *Proceedings of the 4th ACM international symposium on Mobile Ad hoc Networking & Computing*. ACM, 2003, pp. 25–34.

[24] S. Jin and L. Wang, "Content and service replication strategies in multihop wireless mesh networks," in *Proceedings of the 8th ACM international symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2005, pp. 79–86.

[25] L. Yin and G. Cao, "Supporting cooperative caching in ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 1, pp. 77–89, 2006.

[26] M. Mitici, J. Goseling, M. de Graaf, and R. J. Boucherie, "Deployment vs. data retrieval costs for caches in the plane," 2014, to appear in IEEE Wireless Communications Letters.

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

[28] ——, "Social and spatial proactive caching for mobile data offloading," in *Communications Workshops (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 581–586.

[29] B. Blaszczyszyn and A. Giovanidis, "Optimal geographic caching in cellular networks," in *Communications (ICC), 2015 IEEE International Conference on*, June 2015, pp. 3358–3363.

[30] A. Mahanti, N. Carlsson, M. Arlitt, and C. Williamson, "A tale of the tails: Power-laws in internet measurements," *Network, IEEE*, vol. 27, no. 1, pp. 59–64, 2013.

[31] C. Fragouli and E. Soljanin, "Network coding fundamentals," *Foundations and Trends® in Networking*, vol. 2, no. 1, pp. 1–133, 2007.

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

[33] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for lru cache performance," in *Proceedings of the 24th International Teletraffic Congress*. International Teletraffic Congress, 2012, p. 8.

[34] OpenMobileNetwork, `http://map.openmobilenetwork.org/`.

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

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