Optimizing Caching and Recommendation Towards User Satisfaction

Kaiqiang Qi, Binqiang Chen, Chenyang Yang, and Shengqian Han
Beihang University, Beijing, China
Email: {kaiqiangqi,chenbq,cyyang,sqhan}@buaa.edu.cn

***Abstract*—Caching at the wireless edge can improve the user experience in transmission, while personalized recommendation is targeted at satisfying users with contents and can also shape the user demands. In this paper, we jointly optimize caching and recommendation for helpers randomly deployed in cellular networks towards both transmission and content satisfaction. We first introduce a model to reflect the impact of the position of a recommended file in the recommendation list on its request probability. Then, we optimize probabilistic caching policy and personalized recommendation policy to maximize the successful offloading probability under the constraint that the ratings of the recommended files exceed a threshold. We develop an alternating algorithm to find an optimal solution. Simulation and numerical results show evident performance gain over existing relevant policies for both content satisfaction and transmission quality.**

I. Introduction

Caching at the wireless edge, say at base stations (BSs), helpers [1], and users, is expected to improve network performance as well as user experience in terms of transmission quality [1–3]. By optimizing caching policy with known content popularity or user preference, the average download delay [1] or the maximal download delay [4] can be minimized, and the successful offloading probability [5], average ergodic rate [6], or cache-hit probability [7] can be maximized.

Due to the small population and user mobility at the wireless edge, content popularity (i.e., the request probabilities for contents from all users in a small region such as a cell [1, 5–7]) is hard to predict. As a consequence, the gain from wireless edge caching is inevitably reduced. To boost network performance such as offloading probability or cache-hit probability, one solution is to first predict the preference of each user at service gateway in mobile core network, and then aggregate user preference into content popularity for caching policy optimization [2] or optimize caching policy directly with user preference [4, 8]. Another solution is to reduce the uncertainty on user demands by designing recommendation policy friendly to caching [9, 10]. This is inspired by the fact that a byproduct of recommendation is shaping the user demands [11], which has been exploited to achieve higher cache hit rate by content providers. For example, a recommendation list reordering approach was proposed in [12] to increase the likelihood that the already cached contents will be chosen. In [9], a deterministic caching policy at BSs was optimized to maximize the cache hit rate, and a cache-aware recommendation was proposed. Considering that many users may not be determined to request a specific video, a recommendation policy was proposed in [13] to increase soft cache hits by recommending the related contents to each user when the requested content is not cached.

This work is supported by National Natural Science Foundation of China (NSFC) under Grants 61429101 and 61671036.

Nonetheless, improving caching gain by leveraging personalized recommendation calls for the cooperation between content providers and mobile network operators, which fortunately becomes possible recently. One example of the cooperation is between China mobile and Alibaba (who has acquired Youku, a video service provider in China). Traditionally, recommender systems and wireless systems are designed independently. Personalized recommendation is applied to relieve users from information overload by providing the most preferred contents to users [11], with gathered and predicted ratings of each user. Wireless edge caching is designed to boost network performance and improve user experienced transmission quality, with predicted user demands. The cooperation between the two parties can be justified at least partially if recommendation and caching policies can be optimized to satisfy the users with both recommended contents and transmission quality, since user adhesiveness can be improved for both two parties.

In this paper, we attempt to jointly optimize caching and recommendation policies to improve both transmission and content satisfaction for users. To illustrate the gain of the joint optimization, we take an interference-limited network with randomly deployed helpers with cache as an example, employ the integer-valued rating scales to reflect user preference, and employ a required signal-to-interference ratio (SIR) threshold to reflect user satisfaction for transmission. We consider the request probabilities before and after recommendation. We introduce a model to reflect the impact of the position of a content in the recommendation list on the request probability. To ensure content satisfaction, only the files (i.e., contents) with ratings exceeding a threshold will be recommended. To improve transmission satisfaction, we maximize successful offloading probability, i.e., the probability that a requested file can be downloaded from cache with receive SIR exceeding the threshold. We develop an alternating optimization algorithm to find a solution, and use synthesized data to evaluate the performance of the proposed solution.

II. System Model

Consider a cache-enabled network, where BSs are without cache and connected to the core network with backhaul, and helpers are equipped with caches but without backhaul. There are N_u users in a given region, which may request the files from the library with N_f files. Assume that each file is with same file size, but the results are applicable for general case with different sizes by dividing each file into chunks of approximately equal size. Each helper is with cache size of N_c files. The locations of the helpers follow Poisson Point Process with intensity λ_0. To accommodate the location uncertainty of mobile users when optimizing the caching policy, assume that the users are uniformly located in the region. Each helper and each user are equipped with a single antenna.

To maximize the offloaded traffic from helpers, each user is associated with the nearest helper that caches the requested file. If the requested file cannot be found at helpers, the user will be served by the nearest BS to the user.

We consider a probabilistic caching policy, where each helper independently caches files from a content library $\mathcal{F} = \{F'_1, \cdots, F'_f, \cdots, F'_{N_f}\}$ according to an optimized probability distribution, where F'_f is the fth file in set $\mathcal{F}$. Denote the caching probability as $\mathbf{c} = [c_f]_{f=1,\cdots,N_f}$, where $0 \le c_f \le 1$.

We consider a personalized recommendation policy, which recommends M files from $\mathcal{F}$ to each user with ratings no less than r_0, where $M \ll N_f$ and r_0 is a threshold for shaping the user demands. Denote the file set that matches the taste of the kth user adequately as $\mathcal{F}_k = \{F'_f | r_{f|k} \ge r_0\}$. The recommendation list to the kth user can be expressed as a set $\mathcal{M}_k = \{F_1, \cdots, F_i, \cdots, F_M\}$, where $\mathcal{M}_k \subset \mathcal{F}_k \subset \mathcal{F}$ and F_1 may not be the same as F'_1 due to different orders of the elements in $\mathcal{F}$ and $\mathcal{M}_k$. After jointly optimized with the caching policy, such recommendation policy can improve user's satisfaction for transmission by recommending some cached files without compromising the user's satisfaction for contents. Denote the recommendation policy as $\mathbf{X} = [x_{i,f,k}]_{i=1,\cdots,M, f=1,\cdots,N_f, k=1,\cdots,N_u}$, where $x_{i,f,k} = 1$ if the fth file in $\mathcal{F}$ is recommended to the kth user and is located in the ith position of $\mathcal{M}_k$ (i.e., F'_f is the same as F_i), and $x_{i,f,k} = 0$ otherwise.

While the recommendation list is updated after a user sends a request in some recommender systems, the caching and recommendation policies to be jointly optimized are updated simultaneously during a given period (say each day).

III. User Preference and Request Probability

In the literature of wireless edge caching, user preference is defined as the request probability for contents from each user [4, 8, 9]. In the literature of recommendation problem, user preference is expressed by rating, which takes different forms such as real- or integer-valued rating scales (e.g., 1-5 stars) or binary scales (like/dislike) [11]. The request probability from an individual user is assumed as identical to the user preference in [4, 8, 9]. This is reasonable for caching, which is concerned with the number of requests. Yet this assumption may not be true for recommendation, which is concerned with the taste of each user. Considering various information propagation mechanisms, such as word-of-mouth, advertisement, and recommendation, the request probability of a user is not always consistent with the user preference. A user not requesting a content may because the user dislikes or simply is unaware of the content [14]. On the other hand, a user may request a popular content (say a movie) with high probability, but after watching, the user may give a low rating score due to disliking it. In order to jointly optimize caching and recommendation, we differentiate user preference and request probability in this work.

The *user preference*, say the ratings provided by the kth user, is denoted as $\mathbf{r}_k = [r_{f|k}]_{f=1,\cdots,N_f}$, where $r_{f|k} \in \{1, 2, 3, 4, 5\}$ is the rating of the kth user to the fth file in a content library.

The *request probability before recommendation* of the kth user is denoted as $\mathbf{q}_k^{\mathrm{b}} = [q_{f|k}^{\mathrm{b}}]_{f=1,\cdots,N_f}$.

The *request probability after recommendation* of the kth user is denoted as $\mathbf{q}_k^{\mathrm{a}} = [q_{f|k}^{\mathrm{a}}]_{f=1,\cdots,N_f}$. Recommender systems can shape the user demands, i.e., affecting user behaviour in requesting files, but do not change the user preference. Whether or not a user will accept a recommendation depends on how adequately the recommended files match the user preference, and also depends on if the user has determined to request a file. The request probability for a file in the recommendation list further depends on the position of the file in the list [15].

Denote the probability that the kth user accepts a recommendation (i.e., requests a file from $\mathcal{M}_k$) as ξ_k, and the probability that the user requests a file in the ith position of the list conditioned on accepting the recommendation as a_{ki}. If the user accepts the recommendation, then the request probability for the fth file is $q'_{f|k} = \sum_{i=1}^{M} a_{ki} x_{i,f,k}$. Otherwise, the user ignores the recommendation (with probability $1 - \xi_k$) and requests the file with probability $q_{f|k}^{\mathrm{b}}$. Then, the *request probability after recommendation* of the kth user for the fth file in $\mathcal{F}$ can be expressed as

$$\begin{aligned} q_{f|k}^{\mathrm{a}}(\mathbf{X}) &= \xi_k q'_{f|k} + (1 - \xi_k) q_{f|k}^{\mathrm{b}} \\ &= \xi_k \sum_{i=1}^{M} a_{ki} x_{i,f,k} + (1 - \xi_k) q_{f|k}^{\mathrm{b}}. \end{aligned} \tag{1}$$

In such a model, the impact of the file position in the list is considered.[1] This is different from the assumption in [9] that a user uniformly requests a file from the recommendation list (i.e., $a_{ki} = 1/M, i = 1, \cdots, M$).

It is worthy to note that the request probability may not be *correlated with* the user preference. For instance, a user may request a content with high probability but eventually gives a low score after watching.

In practice, user preference can be predicted by collaborative filtering [11], and the request probability can be learned by probabilistic latent semantic analysis [8], say at the service

[1]According to the analysis from online experiment in [15], a_{ki} follows Zipf distribution, i.e., $a_{ki} = i^{-\beta_k} / \sum_{j=1}^{M} j^{-\beta_k}$, where β_k models the skewness of the kth user in requesting files in different positions of $\mathcal{M}_k$.

gateway. The probability that a user accepts a recommendation and the probability that a user requests a file in a particular position of the recommendation list can also be learned from the historical request records of the user. To demonstrate the caching gain from reducing the uncertainty on the request probability of each user, we assume that $\mathbf{r}_k$, $\mathbf{q}_k^{\mathrm{b}}$, ξ_k and a_{ki} are known in the sequel, and leave the issues about how to learn these information in future work.

IV. Joint Caching and Recommendation

In this section, we jointly optimize the probabilistic caching policy and personalized recommendation policy aimed at maximizing the successful offloading probability.

A. Problem Formulation

The successful offloading probability in interference-limited networks is defined as the probability that a requested file from a user can be downloaded from a helper with received SIR larger than a threshold γ_0 required by the user. It can be expressed as

$$p_{off} = \sum_{k=1}^{N_u} w_k \sum_{f=1}^{N_f} q_{f|k}^{\mathrm{a}} \int_0^\infty g_f(r)\mathbb{P}(\gamma_{f,r} > \gamma_0)\mathrm{d}r, \tag{2}$$

where w_k is the probability that a request is sent from the kth user (i.e., the activity level of the user [8]), $g_f(r)$ is the probability density function of the distance r between the user requesting the fth file and the associated helper, and $\gamma_{f,r}$ is the SIR at the user when receiving the fth file. For the considered helper location distribution, $g_f(r) = 2\pi r\lambda_f e^{-\lambda_f \pi r^2}$, and $\lambda_f = \lambda_0 c_f$ is the density of the helpers that cache the fth file.

Similar to the derivation in [16], for Rayleigh fading channel, we can obtain $\mathbb{P}(\gamma_{f,r} > \gamma_0) = \exp(-\pi r^2 \gamma_0^{2/\alpha}(\lambda_0\epsilon_0 - \lambda_f\epsilon_1))$, where $\epsilon_0 = \int_0^\infty \frac{1}{1+u^{\alpha/2}}\mathrm{d}u$, $\epsilon_1 = \int_0^{\gamma_0^{-2/\alpha}} \frac{1}{1+u^{\alpha/2}}\mathrm{d}u$, and α is the path-loss exponent.

By substituting $\mathbb{P}(\gamma_{f,r} > \gamma_0)$ into (2), the successful offloading probability can be derived as

$$p_{off}(\mathbf{c},\mathbf{X}) = \sum_{k=1}^{N_u} w_k \sum_{f=1}^{N_f} q_{f|k}^{\mathrm{a}}(\mathbf{X}) \frac{c_f}{c_f(1-\epsilon_1\gamma_0^{2/\alpha}) + \epsilon_0\gamma_0^{2/\alpha}}, \tag{3}$$

where $1-\epsilon_1\gamma_0^{2/\alpha} = 1 - \int_0^1 \frac{\gamma_0}{\gamma_0 + v^{\alpha/2}}\mathrm{d}v > 1 - \int_0^1 1\mathrm{d}v = 0$.

The problem of jointly optimizing caching policy $\mathbf{c}$ and recommendation policy $\mathbf{X}$ to maximize the successful offloading probability can be formulated as

$$\mathbf{P1}: \max_{\mathbf{c},\mathbf{X}} \quad p_{off}(\mathbf{c},\mathbf{X}) \tag{4}$$

$$s.t. \quad \sum_{f=1}^{N_f} c_f \le N_c, 0 \le c_f \le 1, \tag{4a}$$

$$\sum_{i=1}^{M} x_{i,f,k} \le 1, x_{i,f,k} \in \{0,1\}, \tag{4b}$$

$$\sum_{f=1}^{N_f} x_{i,f,k} \le 1, \tag{4c}$$

$$x_{i,f,k} = 0, \text{ if } r_{f|k} < r_0, \tag{4d}$$

$$F_f' \in \mathcal{F},\ i = 1,\cdots,M,\ k = 1,\cdots,N_u,$$

where (4a) is the constraint on cache size, (4b) means that a file can be recommended to a user at most once, (4c) indicates that a position in the recommendation list can be assigned to at most one file, and (4d) means that a non-preferred file for a user can not be recommended.

Considering that the term $1-\epsilon_1\gamma_0^{2/\alpha}$ in $p_{off}(\mathbf{c},\mathbf{X})$ is positive, it is not hard to show that problem **P1** is not concave by deriving the Hessian matrix of $p_{off}(\mathbf{c},\mathbf{X})$ with respect to $\mathbf{c}$ and $\mathbf{X}$. A locally optimal solution can be found by using interior point method [17], which is with computational complexity of $\mathcal{O}\left((N_f(N_u+1))^{3.5}\right)$. When the size of library and the number of users are large, such complexity is not affordable.

B. An Alternating Optimization Algorithm

In what follows, we propose a low-complexity algorithm to find the solution of problem **P1**.

First, we show that once the caching policy is given, the optimal recommendation policy can be obtained. Then, we show that once the recommendation policy is given, the optimal caching policy can be obtained.

Proposition 1. *With any given caching policy* $\mathbf{c}$*, the optimal recommendation policy for the* k*th user is*

$$x_{i,f,k}^* = \begin{cases} 1, & \text{if } F_f' = F_i, \\ 0, & \text{if } F_f' \ne F_i, \end{cases} \tag{5}$$

where each file in recommendation list $\mathcal{M}_k$ *is obtained as*

$$\begin{aligned} F_1 &= \operatorname*{argmax}_{j\in\mathcal{F}_k} c_j, \\ F_2 &= \operatorname*{argmax}_{j\in\mathcal{F}_k\backslash F_1} c_j, \\ &\vdots \\ F_M &= \operatorname*{argmax}_{j\in\mathcal{F}_k\backslash\{F_1,\cdots,F_{M-1}\}} c_j. \end{aligned} \tag{6}$$

Proof: See Appendix A. ■

In order not to deviate from the goal of recommender systems, i.e., relieving users from information overload, M is usually small. Then, the number of files in $\mathcal{F}_k$ is usually larger than M, which is true even when r_0 is set as 5 (as shown in the dataset later). The proposition suggests that the optimal recommendation policy for a given caching policy is to select the files from $\mathcal{F}_k$ with the maximal M values of caching probability. Besides, the file with the highest caching probability (i.e., F_1) ranks in the first position in the list.

Remark: If there is no constraint on the content satisfaction, i.e., $r_0 = 1$ such that $\mathcal{F}_k = \mathcal{F}, \forall k = 1,\cdots,N_u$, the recommendation list will be identical for all users as in [10]. Then, the policy becomes non-personalized.

With any given recommendation policy $\mathbf{X}$, it is not hard to prove that $p_{off}(\mathbf{c}, \mathbf{X})$ is concave in $\mathbf{c}$. Then, the optimal caching policy can be derived from the Karush-Kuhn-Tucker conditions as

$$c_f^* = \left[\frac{1}{1 - \epsilon_1 \gamma_0^{2/\alpha}} \left(\left(\frac{\epsilon_0 \gamma_0^{2/\alpha} \tilde{p}_f(\mathbf{X})}{\mu} \right)^{1/2} - \epsilon_0 \gamma_0^{2/\alpha} \right) \right]_0^1, \tag{7}$$

where $[x]_0^1 = \min\{\max\{x, 0\}, 1\}$ denotes that x is truncated by 0 and 1, μ can be solved from $\sum_{f=1}^{N_f} c_f^* = N_c$ by bisection search, and $\tilde{p}_f(\mathbf{X}) = \sum_{k=1}^{N_u} w_k q_{f|k}^{\mathrm{a}}(\mathbf{X})$ is the content popularity of the fth file after recommendation.

According to previous analysis, we can resort to alternating optimization algorithm to find a locally optimal solution of problem **P1**, which improves the offloading gain at each iteration by alternatively optimizing the recommendation and caching policies until the algorithm converges. We can first optimize the recommendation policy for a given caching policy. The procedure is shown in **Algorithm 1**.

Algorithm 1 The alternating optimization algorithm.

```
1: Initialize caching policy $\mathbf{c}$. $\epsilon$ is a predetermined value.
2: repeat
3:     Given $\mathbf{c}$, $\mathbf{X}'$ is obtained from (5).
4:     Given $\mathbf{X}'$, $\mathbf{c}'$ is obtained from (7).
5:     Compute $\Delta = p_{off}(\mathbf{c}', \mathbf{X}') - p_{off}(\mathbf{c}, \mathbf{X}')$ by (3).
6:     $\mathbf{c} \leftarrow \mathbf{c}', \mathbf{X} \leftarrow \mathbf{X}'$.
7: until $\Delta < \epsilon$
8: return $\mathbf{c}^* = \mathbf{c}, \mathbf{X}^* = \mathbf{X}$.
```

We can also first optimize the caching policy for a given recommendation policy. Since problem **P1** is not concave, we can use different initializations and pick the best solution to increase the opportunity to find the global optimal solution. All the initializations can converge after only two iterations when $\epsilon = 10^{-2}$, as shown by numerical results. When ϵ is set as a less value, more iterations are required for convergence.

V. Numerical and Simulation Results

In this section, we evaluate the performance of joint caching and recommendation policy and analyze the impact of several key factors. We consider two performance metrics, respectively from the perspective of mobile operators and content providers. The first metric is successful offloading probability, where γ_0 can reflect user experience in transmission. The second metric is user satisfaction probability, defined as the *probability that the request of a user is served by a helper with SIR larger than* γ_0 *and the ratings of the recommended files are larger than a value* $\bar{r}_0$, which can reflect both transmission and content satisfaction.[2] $\bar{r}_0$ is not always the same as r_0.

[2]$\bar{r}_0$ can reflect the real user satisfaction for contents. If the ratings of the recommended files of a user exceed $\bar{r}_0$, then the user is satisfied with the recommendation.

We compare the policy obtained from the alternating algorithm (with legend "*Proposed-Joint*") with two baselines:

- "*Existing-Joint*": This is the strategy proposed in [9]. The strategy first optimizes a deterministic caching policy to maximize the cache hit rate assuming known request probability after recommending the top-M files to each user according to the user preference (i.e., rating). Then, the strategy adjusts the recommended files according to the cached files, and finally shows the adjusted recommendation list to each user.
- "*Separate*": This is the strategy where caching and recommendation policies are designed separately, which reflects the common practice. The strategy optimizes a probabilistic caching policy that maximizes the successful offloading probability according to the request probability before recommendation,[3] and recommends the top-M files according to the user preference.

In many real datasets available in the literature, either only the ratings given by each user (say *MovieLens* [18]) or only the numbers of requests from each user (say *Million Songs* [19]) are recorded. Different datasets correspond to different scenarios (say different collections of users), reflecting different user behaviors (e.g., content popularity, user similarity). Under the same content popularity, which is the weighted average of the request probabilities of multiple users, the similarity among the request probability of each user has large impact on the caching gain [8]. To investigate the impact of request probability and user preference on the performance for systems serving different groups of users, we synthesize the request probability and user preference. The ratings in *MovieLens* dataset are integer, which will lead to coarse-grained quantization of request probability if a rating is translated to the probability. Therefore, we first synthesize request probability before recommendation, and then synthesize user preference.

The *request probability* and *activity level* of each user are synthesized using the method in [8] for given content popularity and user similarity in request probability.

To capture the relation between request probability and user preference, we consider two approaches to synthesize *user preference* from the request probabilities. One is assuming that user preference is correlated with request probability, i.e., the rating is high when the request probability is large. As shown in Fig. 1, we first sort the files according to the request probabilities of each user in descending order, and then divide the corresponding files into five subsets according to the rating distribution (i.e., the probabilities that the five ratings are given by all users). In particular, the files in the first subset are with highest request probabilities and hence they are with rating of 5, the files in the second subset are with rating of 4, and so on. The other approach reflects an extreme case where user

[3]In practice, the reason of a user requesting a file may be that the user knows something about the file from friends or advertisement, and may also be that the user accepts a recommendation. For the latter case, the strategy is to optimize the caching policy according to the request probability after recommendation. Simulation results show that such a strategy outperforms the "*Separate*" strategy slightly only when ξ_k is high.

preference is irrelevant to request probability, i.e., the ratings are random variables following the rating distribution.

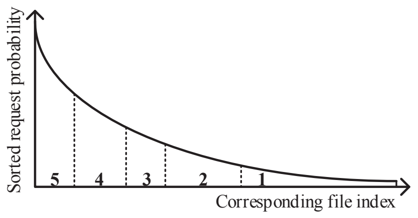


Fig. 1. Synthesizing user preference from request probability.

The rating distribution can be obtained from real datasets. In [20], the rating distribution is analyzed for several datasets, among which *MovieLens 1M* and *Netflix* are with similar distribution. Table I provides the rating distribution in the original *MovieLens 1M* dataset (the second row).

TABLE I
ORIGINAL AND COMPLETED RATING DISTRIBUTION

Ratings	1	2	3	4	5
Original	0.056	0.108	0.261	0.349	0.226
Complete	0.13	0.138	0.381	0.315	0.036

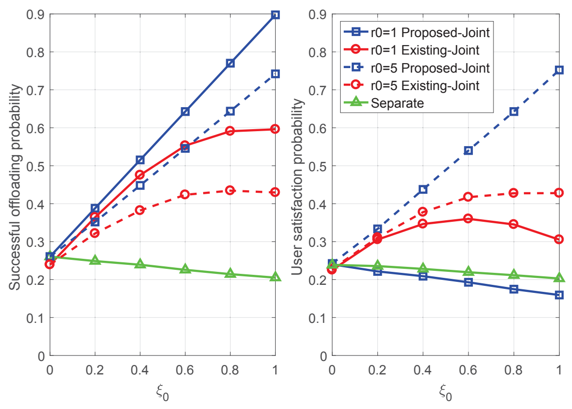
We can observe high probability of the rating of 5. Using such rating distribution to synthesize user preference will yield optimistic results. This is because a user usually gives rating to a content after the user has watched the content, and a user usually requests a content if the user expects to like it. In real datasets, the ratings provided by users are extremely sparse, where there exists massive missing data (i.e., the data samples without ratings). The missing data may be attributed to dislike of the users (i.e., negative ratings), or simply owing to unawareness of the contents (i.e., unknown ratings) [14]. As a result, the rating distribution obtained from the original dataset may deviate from the real distribution. To deal with this problem, we use matrix factorization technique in collaborative filtering to predict the missing ratings [11], where the predicted values are rounded to 1$\sim$5. Table I shows the rating distribution of all ratings included the predicted ratings (the third row), which is employed to synthesize user preference.

We consider a circular area with radius 500 m and $N_u = 100$ users.[4] The density of the helper is $\lambda_0 = 50/(500^2\pi)$ m^{-2}. The path-loss exponent is $\alpha = 3.7$. The SIR threshold is $\gamma_0 = -10$ dB, which corresponds to 2.75 Mbps with the transmission bandwidth 20 MHz. The content satisfaction threshold is $r_0 = 5$. We also provide the results for $r_0 = 1$ to show what happens if user preference is not taken into account in the optimization. The library size is $N_f = 1000$. $N_c = 50$ files can be cached at each helper, and $M = 10$ files are recommended to each user. The content popularity is Zipf distribution with skewness parameter 0.6.

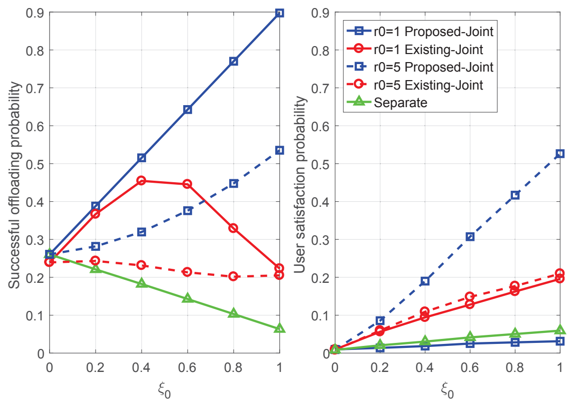
[4]Considering that the users are with different activity levels and only the requests able to be successfully offloaded are served by the helpers, each helper will serve at most one user in the setup.

The user similarity in request probability is $s = 0.3$. The probability of requesting files from recommendation list is set as $\xi_k = \xi_0 = 0.5$. a_{ki} follows Zipf distribution, and $\beta_k = 0.6$. The pre-determined value, ϵ, in Algorithm 1 is set as 10^{-6}. All numerical and simulation results are averaged over 20 tests. In each test, the user preference, request probability, and activity level are synthesized independently. Unless otherwise specified, these parameters are used in the sequel.

The successful offloading probability is evaluated from numerical results, which are computed from (3) with $\mathbf{c}$ and $\mathbf{X}$. The user satisfaction probability is evaluated by simulations results obtained from 20 tests, each consisting of 20 Monte Carlo trials. In each trial, the locations of each helper and each user, and the channel coefficients change independently. The cached files in each helper change according to the optimized caching probability using the method proposed in [7], and 200 requests are generated according to the activity level and request probability after recommendation of each user. The value reflecting the real content satisfaction is set $\bar{r}_0 = 5$, i.e., each user will be satisfied with the recommendation only if the recommended files are with rating of 5.



(a) User preference is correlated with request probability.



(b) User preference is irrelevant to request probability.

Fig. 2. Impact of the probability of accepting recommendation.

In Fig. 2, we show the impact of the probability of accepting recommendation as well as the relation between user preference and request probability on the two metrics. From the left figures of Figs. 2(a) and (b), we can see that for the successful offloading probability, the proposed policy outperforms two baselines consistently. The performance gain is remarkable for large value of ξ_0, which corresponds to the scenario where users are flexible to request files and the recommended files are preferred by each user. The gain over "*Existing-Joint*" strategy comes from two factors. 1) The proposed solution satisfies the user requirement for transmission quality and user preference for contents. 2) The proposed solution recommends files considering the impact of their positions in the recommendation list. The "*Separate*" strategy performs the worst, because recommendation and caching policies are independent, resulting in the recommended files being less cached. Comparing the left and right figures in Fig. 2(a), we find that the user satisfaction probabilities are very close to the successful offloading probabilities for the proposed solution and "*Existing-Joint*" strategy with $r_0 = 5$ (because $\bar{r}_0$ is also 5), but become inferior to the successful offloading probabilities when $r_0 = 1$. The user satisfaction probability of the proposed policy deteriorates more severely for $r_0 = 1$. This is because when setting $r_0 = 1$ in the optimization, the proposed policy only recommends the popular contents without considering user preference, while "*Existing-Joint*" strategy attempts to recommend some cached files with higher ratings. Comparing Figs. 2(a) with (b), we find that the performance will degrade if the user preference becomes irrelevant to the request probability, except the successful offloading probability achieved by the proposed solution with $r_0 = 1$.

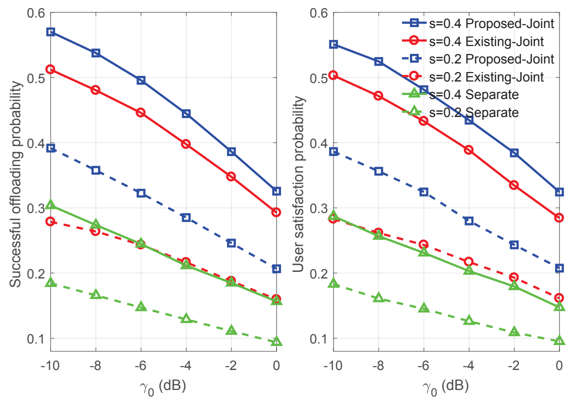


Fig. 3. Impact of the SIR threshold and request similarity, $r_0 = 5$.

In Fig. 3, we show the impact of γ_0, user similarity in request probability (denoted as s) and different values of ξ_k. We set $\xi_k \sim [0, 1]$, i.e., the probability of accepting recommendation by each user is a uniformly distributed random variable. The maximal SIR threshold, 0 dB, corresponds to 20 Mbps with the transmission bandwidth 20 MHz. User preference is correlated with request probability. The results in the left and right figures are close since $\bar{r}_0 = r_0 = 5$, which show that the proposed solution outperforms the two baselines. As expected, both the successful offloading probability and user satisfaction probability decrease with the SIR threshold. The performance increases with the user similarity in request probability. This is because when the user demands are less heterogeneous (which implies that the user demands become less uncertain), the content popularity is more skewed.

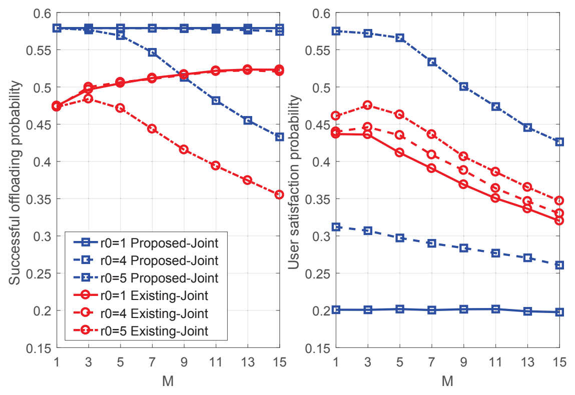


Fig. 4. Impact of the size of recommendation list, $\gamma_0 = -10$ dB.

In Fig. 4, we show the impact of the size of recommendation list. User preference is correlated with request probability. When $r_0 = 5$, the performance of the proposed solution decreases with M, because the recommendation policy becomes more personalized to users. Then, the user demands will be more heterogeneous if the list is long, and hence the content popularity after recommendation will become less skewed, resulting in the deteriorated performance. When $r_0 = 1$, the proposed solution always recommends the same list of files to all users as discussed in the remark, and almost recommends same files to all users when $r_0 = 4$ because there are more than 300 files with ratings of 4. Since $M < N_c$, the content popularity of the $N_f - N_c$ less popular files does not change with M, and hence the successful offloading probability almost does not change with M when $r_0 = 1$ and 4. For the proposed solution with $r_0 = 4$, the successful offloading probability is high for all values of M but the user satisfaction probability is low. This indicates that shaping user demands can improve transmission experience but may sacrifice overall user satisfaction. However, when setting $r_0 = 5$, the proposed solution can achieve almost the same performance for both metrics. There exist optimal values of M for both metrics achieved by "*Existing-Joint*" strategy (also true for the successful offloading probability when $r_0 = 1$ and 4 if M is larger than 15, not shown in the figure). This is because this strategy tends to recommend the cached files with high ratings when M is small. As M grows, a long recommendation list may contain more cached files, where more files are with ratings less than $\bar{r}_0$ (hence no longer makes users satisfied). However, if M is too large, the user demands will become more heterogeneous, resulting in the deteriorating performance.

VI. Conclusions

In this paper, we jointly optimized caching and recommendation to improve the network performance without compromising the transmission quality and content satisfaction of each user. We employed SIR threshold to reflect user requirement for transmission quality and ratings to reflect user satisfaction for contents. We considered the impact of the file position in recommendation list on the request probability. We formulated a joint caching and recommendation problem for an interference-limited network and developed an alternating optimization algorithm to find the caching and recommendation policies. Simulation and numerical results showed that the proposed solution outperforms the existing policies in terms of both successful offloading probability and user satisfaction probability. The preliminary results suggest that both mobile network operators and content providers can benefit from the cooperation by the joint optimization.

Appendix A
Proof of Proposition 1

Since the recommendation policy is personalized for each user, when caching policy $\mathbf{c}$ is given, problem **P1** can be split into N_u sub-problems. In particular, the optimal recommendation policy to the kth user, $\mathbf{X}_k^* = [x_{i,f,k}^*]$, can be obtained from the following problem

$$\mathbf{P2}: \max_{\mathbf{X}_k} \quad w_k \sum_{f=1}^{N_f} q_{f|k}^{\mathrm{a}}(\mathbf{X}_k) \frac{c_f}{c_f(1-\epsilon_1\gamma_0^{2/\alpha}) + \epsilon_0\gamma_0^{2/\alpha}} \tag{A.1}$$

$$s.t. \quad \text{(4b), (4c), (4d)}, F_f' \in \mathcal{F}, i = 1, \cdots, M.$$

Substituting (1) into the objective function of problem **P2**, we can see that w_k, ξ_k and $q_{f|k}^{\mathrm{b}}$ do not affect the solution of the problem. Then, the objective function degenerates into $\sum_{f=1}^{N_f}\sum_{i=1}^{M} \frac{a_{ki}c_f x_{i,f,k}}{c_f(1-\epsilon_1\gamma_0^{2/\alpha})+\epsilon_0\gamma_0^{2/\alpha}} = \sum_{i=1}^{M} a_{ki} \sum_{f=1}^{N_f} \frac{c_f}{c_f(1-\epsilon_1\gamma_0^{2/\alpha})+\epsilon_0\gamma_0^{2/\alpha}} x_{i,f,k}$, which can be shown as monotonically increasing with c_f by examining its first derivative with respect to c_f.

Further considering that a_{ki} and $\frac{c_f}{c_f(1-\epsilon_1\gamma_0^{2/\alpha})+\epsilon_0\gamma_0^{2/\alpha}}$ are decoupled in the objective function and a_{ki} determines the position of each recommended file, we can obtain the optimal recommendation policy as follows. First, we find the file indices with the maximal M values of caching probability in $\mathcal{F}_k$ to constitute the recommendation list. Then, we successively assign the recommended files to corresponding positions in the list. According to the rearrangement inequality [21], to maximize the objective function, the file with maximal value of c_f should occupy the position with maximal value of a_{ki}. This completes the proof.

References

[1] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[2] E. Zeydan, E. Bastug, M. Bennis, M. A. Kader, I. A. Karatepe, A. S. Er, and M. Debbah, "Big data caching for networking: Moving from cloud to edge," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 36–42, Sept. 2016.

[3] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: Design aspects, challenges, and future directions," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 22–28, Sep. 2016.

[4] D. Liu, C. Yang, and V. Leung, "When exploiting individual user preference is beneficial for caching at base stations," in *Proc. IEEE ICC Workshops*, 2018.

[5] J. Rao, H. Feng, C. Yang, Z. Chen, and B. Xia, "Optimal caching placement for D2D assisted wireless caching networks," in *Proc. IEEE ICC*, 2016.

[6] X. Xu and M. Tao, "Modeling, analysis, and optimization of coded caching in small-cell networks," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3415–3428, Aug. 2017.

[7] B. Blaszczyszyn and A. Giovanidis, "Optimal geographic caching in cellular networks," *in Proc. IEEE ICC*, 2015.

[8] B. Chen and C. Yang, "Caching policy optimization for D2D communications by learning user preference," in *Proc. IEEE VTC Spring*, 2017.

[9] L. E. Chatzieleftheriou, M. Karaliopoulos, and I. Koutsopoulos, "Caching-aware recommendations: Nudging user preferences towards better caching performance," in *Proc. IEEE INFOCOM*, 2017.

[10] K. Guo, C. Yang, and T. Liu, "Caching in base station with recommendation via Q-learning," in *Proc. IEEE WCNC*, 2017.

[11] M. D. Ekstrand, J. T. Riedl, J. A. Konstan *et al.*, "Collaborative filtering recommender systems," *Foundations and Trends® in Human–Computer Interaction*, vol. 4, no. 2, pp. 81–173, 2011.

[12] D. K. Krishnappa, M. Zink, C. Griwodz, and P. Halvorsen, "Cache-centric video recommendation: An approach to improve the efficiency of YouTube caches," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 11, no. 4, p. 48, 2015.

[13] P. Sermpezis, T. Spyropoulos, L. Vigneri, and T. Giannakas, "Femto-caching with soft cache hits: Improving performance with related content recommendation," in *Proc. IEEE CLOBECOM*, 2017.

[14] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *Proc. IEEE ICDM*, 2008.

[15] R. Zhou, S. Khemmarat, and L. Gao, "The impact of YouTube recommendation system on video views," in *Proc. ACM SIGCOMM*, 2010.

[16] B. Chen, C. Yang, and Z. Xiong, "Optimal caching and scheduling for cache-enabled D2D communications," *IEEE Commun. Lett.*, vol. 21, no. 5, pp. 1155–1158, 2017.

[17] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[18] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interactive Intell. Syst.*, vol. 5, no. 4, p. 19, 2016.

[19] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proc. ISMIR*, 2011. [Online]. Available: https://labrosa.ee.columbia.edu/millionsong/tasteprofile

[20] B. M. Marlin and R. S. Zemel, "Collaborative prediction and ranking with non-random missing data," in *Proc. ACM RecSys*, 2009.

[21] G. H. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*. Cambridge university press, 1988.