Abstract—To deal with the rapid growth of high-speed and/or ultra-low latency data traffic for massive mobile users, fog radio access networks (Fog-RANs) have emerged as a promising architecture for next-generation wireless networks. In Fog-RANs, the edge nodes and user terminals possess storage, computation and communication functionalities to various degrees, which provides high flexibility for network operation, i.e., from fully centralized to fully distributed operation. In this paper, we study the cache placement problem in Fog-RANs, by taking into account physical-layer transmission schemes and diverse content preferences of different users. We develop both centralized and distributed transmission aware cache placement strategies to minimize users’ average download delay subject to the storage capacity constraints. In the centralized mode, the cache placement problem is transformed into a matroid constrained submodular maximization problem, and an approximation algorithm is proposed to find a solution within a constant factor to the optimum. In the distributed mode, a belief propagation based distributed algorithm is proposed to provide a suboptimal solution, with iterative updates at each BS based on locally collected information. Simulation results show that by exploiting caching and cooperation gains, the proposed transmission aware caching algorithms can greatly reduce the users’ average download delay.

Index Terms—Content placement, Fog-RAN, submodular optimization, belief propagation.

I. INTRODUCTION

With the explosive growth of consumer-oriented multimedia applications, a large scale of end devices, such as smart phones, wearable devices and vehicles, need to be connected via wireless networking [2]. This has triggered the rapid increase of high-speed and/or ultra-low latency data traffic that is very likely generated, processed and consumed locally at the edge of wireless networks. To cope with this trend, fog radio access network (Fog-RAN) is emerging as a promising network architecture, in which the storage, computation, and communication functionalities are moved to the edge of wireless networks, i.e., to the near-user edge devices and end-user terminals [3]–[5]. To further improve the delivery rate and decrease latency for mobile users, a promising solution is to push the popular contents towards end users by caching them at the edge nodes in Fog-RANs [3]. Thus, the content delivery service of mobile users consists of two phases, i.e., cache placement and content delivery [1]. This work was supported in part by the NSFC under Grant No. 61601255, the Hong Kong Research Grants Council under Grant No. 610113, the Scientific Research Foundation of Ningbo University under Grant No. 010-421703900 and the Zhejiang Open Foundation of the Most Important Subjects under Grant No. 010-421500212. This work was presented in part at the IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, May 2016 [2].

This work was supported in part by the NSFC under Grant No. 61601255, the Hong Kong Research Grants Council under Grant No. 610113, the Scientific Research Foundation of Ningbo University under Grant No. 010-421703900 and the Zhejiang Open Foundation of the Most Important Subjects under Grant No. 010-421500212. This work was presented in part at the IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, May 2016 [1]. J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.

J. Liu is with the College of Electrical Engineering and Computer Science, Ningbo University, Zhejiang, China, 315211. E-mail: eeliju@nju@gmail.com. B. Bai is the Future Network Theory Lab, Huawei Technologies Co., Ltd., Shatin, N. T., Hong Kong, E-mail: baibao8@huawei.com. J. Zhang and K. B. Letaief are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. K. B. Letaief is also with Hamad bin Khalifa University, Doha, Qatar. E-mail: ee@ust.hk, eekhale@ust.hk.
costs of all the SBSs subject to the cache storage capacities. In [20], the design of optimal cache placement was pursued for wireless networks, by taking the extra delay induced via backhaul links and physical-layer transmissions into consideration. The authors in [20] proposed user preference profile based caching policies for radio access networks along with backhaul and wireless channel scheduler to support more concurrent video sessions. In [22], mobility-aware caching strategies were proposed to exploit user mobility patterns to improve cache performance. The joint routing and caching problem was studied for small-cell networks and heterogeneous networks in [23] and [24], respectively, subject to both the storage and transmission bandwidth capacity constraints on the small-cell BSs.

The existing works mainly focused on designing centralized cache placement strategies for specific network structures (e.g. small cell networks), where some specific transmission schemes are applied for content delivery. However, very few works have studied the cache placement problem in Fog-RANs. We notice that different users may be connected to Fog-RANs in different ways and with different transmission opportunities. Meanwhile, Fog-RANs support flexible network operation, i.e., from fully centralized to fully distributed operation. This motivates us to develop both centralized and distributed transmission aware cache placement strategies for the emerging Fog-RANs so that the spectrum efficiency of content delivery is improved as much as possible.

In this paper, we consider a Fog-RAN system, where each user is served by one or multiple network edge devices, e.g., base stations (BSs), and each BS is equipped with a cache of finite capacity. In contrast to [23] and [24] where each user has the same file preference and file delivery scheme, we consider that the users have different file preferences [25] and possibly different candidate transmission schemes. Then, we formulate an optimization problem to minimize the users’ average download delay subject to the BSs’ storage capacities, which turns out to be NP-hard. To deal with this difficulty, we apply different optimization techniques to find efficient cache placement policies for centralized and distributed operation modes of Fog-RANs, respectively.

In the centralized mode, we transform the delay minimization problem into a matroid constrained submodular maximization problem [26]. In this problem, the average delay function is submodular for all the possible transmission schemes, and the cache placement strategy subject to the BSs’ storage capacities is a partition matroid. Based on the submodular optimization theory [26], we then develop a centralized low-complexity algorithm to find a caching solution within $1/2$ of the optimum in polynomial-time complexity $O(MNK)$, where $M$, $N$ and $K$ denote the number of BSs, files and users, respectively.

In the distributed mode, we develop a low-complexity belief propagation based distributed algorithm to find a suboptimal cache placement strategy [27]. Based on local information of its storage capacity, the users in its serving range and their file request statistics, each BS perform individual computation and exchange its belief on the local caching strategy with its neighboring BSs iteratively. Through iterations, the distributed algorithm converges to a suboptimal caching solution which achieves an average delay performance comparable to the centralized algorithm, as shown by simulation results.

As shown in Fig. 1, we consider a Fog-RAN consisting of $M$ edge nodes, i.e., BSs, and $K$ mobile users. Let $A = \{a_1, \ldots, a_M\}$ and $U = \{u_1, \ldots, u_K\}$ denote the BS set and the user set, respectively. Each user can be served by one or multiple BSs, depending on the way it connects to the Fog-RAN. The connectivity between the users and the BSs is denoted by a $K \times M$ matrix $L$, where each binary element $l_{km}$ indicates whether user $u_k$ can be served by BS $a_m$. That is, $l_{km} = 1$ if user $u_k$ is located in the coverage of BS $a_m$, and $l_{km} = 0$ otherwise. The set of users in the coverage of BS $a_m$ is denoted by $U_m = \{u_k \in U | l_{km} = 1\}$. Similarly, the set of serving BSs of user $u_k$ is denoted by $A_k = \{a_m \in A | l_{km} = 1\}$.

Suppose that the library of $N$ files, denoted by $F = \{f_1, \ldots, f_N\}$, is stored at one or multiple content servers.
which could be far away in the cloud data center. The content servers can be accessed by the BSs via backhaul links, as illustrated in Fig. 1. Assume all the files have the same size, i.e., \( |f_n| = |f| \) (\( \forall f_n \in F \)). The file popularity distribution conditioned on the event that user \( u_k \) makes a request is denoted by \( p_{uk} \), which can be viewed as the user preference indicator and estimated via some learning procedure [28]. The user’s file preferences are normalized such that \( \sum_{n=1}^{N} p_{nk} = 1 \). We also assume that each BS \( a_m \) has a finite-capacity storage. Denote by \( Q_m \) the normalized storage capacity of BS \( a_m \), which means that each BS \( a_m \) can store at most \( Q_m \) files. Let \( x_{nm} \) be a binary variable indicating whether file \( f_n \) is cached at BS \( a_m \). That is, \( x_{nm} = 1 \) if file \( f_n \) is stored at BS \( a_m \), and otherwise \( x_{nm} = 0 \). The caching variables \( \{x_{nm}\} \) shall be determined collaboratively by the BSs to improve the probability that the users’ requested files can be found in the caches of the BSs, i.e., the hit probability. Meanwhile, the cooperative caching strategy, denoted by \( X \), should also be carefully designed to provide flexible and cooperative transmission opportunities for each user.

When user \( u_k \) makes a request for file \( f_n \), the serving BSs \( A_k \) jointly decide how to transmit to this user based on the caching strategy \( X \). Specifically, when file \( f_n \) is cached in one or multiple BSs, the BSs transmit this file to the user directly by employing some transmission schemes, e.g., non-cooperative transmission or cooperative beamforming, as shown in Fig. 1. When file \( f_n \) has not been cached in any serving BS of the user, the associated BSs \( A_k \) fetch the file from a content server via backhaul links before they transmit to user \( u_k \) over wireless channels.

The users’ file delivery performance depends not only on the cache placement strategy but also on the specific transmission schemes applied to deliver the files to the users. In the following, we discuss the file delivery rates for some typical physical-layer transmission schemes, when the requested file is cached in one or multiple associated BSs.

1) Non-cooperative Transmission: When user \( u_k \) is served by one single BS \( a_m \), a non-cooperative transmission scheme be applied by this BS to transmit the file to the user directly, if the requested file \( f_n \) is cached in this BS. Assume that efficient interference management schemes are applied and interference power is constrained by a fixed value \( \chi \). Let \( \text{SINR}_m = \frac{P_m}{N_0} \) denote the target signal-to-interference-plus-noise ratio (SINR) at the transmitter side, where \( P_m \) is the average transmission power at BS \( a_m \), \( N_0 \) is the power spectral density of noise, and \( B \) is the system bandwidth. The file delivery rate in time slot \( i \) can be estimated as

\[
R_{nk}(X, i) = B \log \left( 1 + \frac{|h_{km}(i)|^2 l_{km} x_{nm} \text{SINR}_m}{} \right),
\]

where \( h_{km}(i) \) denotes the channel coefficient between user \( u_k \) and BS \( a_m \) in time slot \( i \).

2) Cooperative Beamforming: When user \( u_k \) is served by multiple BSs, cooperative beamforming can be applied by the associated BSs \( A_k \), if file \( f_n \) has been cached in multiple BSs and the instantaneous channel state information is available. During the file delivery phase, cooperative beamformer can be created possibly in a distributed way to avoid signaling overhead [39]. Accordingly, the file delivery rate in time slot \( i \) is estimated as

\[
R_{nk}(X, i) = B \log \left( 1 + \sum_{a_m \in A_k} |h_{km}(i)|^2 x_{nm} \text{SINR}_m \right),
\]

where \( A_{kn} \subseteq A_k \) denotes a set of BSs that transmit file \( f_n \) to user \( u_k \) via cooperative beamforming.

In this work, we aim at finding the optimal cache placement strategy to minimize the average download delay, considering different candidate transmission schemes for each user, as be presented in the next section.

III. PROBLEM FORMULATION FOR CACHE PLACEMENT

In this section, we first show how to calculate the average download delay by applying martingale theory [31]. Then, we formulate the cache placement problem. Let \( D_{nk}(X) \) denote the average delay for user \( u_k \) to download file \( f_n \) from its serving BSs for a given caching strategy \( X \) and a specific transmission scheme. When file \( f_n \) has been cached in one or multiple BSs, user \( u_k \) can download this file from the associated BSs with rate \( R_{nk}(X, i) \) (cf. (1)-(2)) in each time slot \( i \). In this case, it takes at least \( T_{nk}(X) \) time slots for user \( u_k \) to successfully receive all the bits of file \( f_n \). The minimum number of time slots \( T_{nk}(X) \) can be evaluated as

\[
T_{nk}(X) = \arg \min \left\{ T : \sum_{i=1}^{T} R_{nk}(X,i) \geq \left\lceil \frac{|f_n|}{\Delta t} \right\rceil \right\},
\]

where \( \Delta t \) is the duration of one time slot. Thus, for user \( u_k \), the average delay of downloading file \( f_n \) is expressed as

\[
D_{nk}(X) = E_{h} \{ T_{nk}(X) \} \Delta t.
\]

When file \( f_n \) has not been cached at any associated BS, one or multiple serving BSs of user \( u_k \), denoted by \( A_k \), should first fetch the file from the content server via the backhaul link before delivering the requested file to this user over wireless channel. Let \( D_{nk} \) denote the extra delay of downloading file \( f_n \) from the content server to the BSs \( A_k \). We then evaluate the average download delay under the assumption that the channel coefficients \( \{ h_{km}(i) \} \) are identically and independently distributed (i.i.d.) across the time slots \( i \) in the following theorem.

Theorem 1. If the channel coefficients \( \{ h_{km}(i) \} \) are i.i.d. across the time slots, the average delay for user \( u_k \) to download file \( f_n \) can be expressed as

\[
D_{nk}(X) = \begin{cases} \frac{|f_n|}{E_h \{ R_{nk}(X) \}} , & \sum_{a_m \in A_k} x_{nm} \neq 0, \\ D_{nk} + \frac{|f_n|}{E_h \{ R_{nk}(X) \}} , & \sum_{a_m \in A_k} x_{nm} = 0. \end{cases}
\]

where \( E_h \{ \cdot \} \) denotes the expectation over the channel coefficients \( \{ h_{km}(i) \} \) and \( X_k \) is a caching strategy with \( x_{nm} = 1 \) for \( a_m \in A_k \).

Proof: The proof is deferred to Appendix A.

From this theorem, we can evaluate the average download delay by (5) for any given caching strategy and employed
transmission scheme. Without loss of generality, we assume that the users’ average delay of downloading file $f_n$ from the content server is larger than the average delay of direct file delivery from the BSs and the following inequality holds:

$$\frac{|f_n|}{E_h \{R_n(X_k)\}} + D_{nk} > \max_{\sum_{a_m \in A_k} x_{nm} \neq 0} \left\{ \frac{|f_n|}{E_h \{R_n(X)\}} \right\}.$$  

(6)

If $D_{nk}$ is much larger than $\frac{|f_n|}{E_h \{R_n(X_k)\}}$, the average delay $D_{nk}(X)$ can be approximated by $D_{nk}$ when $\sum_{a_m \in A_k} x_{nm} = 0$. Notice that $D_{nk}$ is the sum of the delay of file delivery within the Internet which mainly depends on the level of congestion in the network, and the delay of file delivery via backhaul links which may depend on the backhaul capacities and the caching strategy $X$. Considering all these effects, the impact of the caching strategy $X$ on the delay $D_{nk}$ is negligible. Hence, we assume that the average delay $D_{nk}$ is fixed and can be evaluated by the average time of downloading file $f_n$ from the content server to the serving BSs of user $u_k$.

In the considered system, we seek to design transmission aware cache placement strategies to minimize the average delay of all the users, by taking different candidate transmission schemes for each user into consideration. Formally, the cache placement problem can be formulated as follows

$$\min_{\{x_{nm}\}} \bar{D}(X) = \frac{1}{K} \sum_{k=1}^{K} \sum_{n=1}^{N} p_{nk} \bar{D}_{nk}(X)$$  

(7)

subject to

$$\sum_{n=1}^{N} x_{nm} \leq Q_m, \forall a_m \in A, \quad (a)$$

$$x_{nm} \in \{0, 1\}, \forall f_n \in F, a_m \in A, \quad (b)$$

where constraint (7a) means that each BS $a_m$ is allowed to store at most $Q_m$ files. Since the variable $x_{nm}$ is binary, Problem (7) is a constrained integer programming problem, which is generally NP-hard [32]. Hence, it is very challenging to find the optimal solution $X^*$ to Problem (7). In the next two sections, we show how to approach the optimal cache placement strategy in the centralized and distributed modes of Fog-RANs, respectively.

IV. SUBMODULAR OPTIMIZATION BASED CENTRALIZED CACHE PLACEMENT ALGORITHM

As a powerful tool for solving combinatorial optimization problems, the submodular optimization is applied when Fog-RANs operate in the centralized mode with the aid of a central controller. In this section, Problem (7) is first reformulated into a monotone submodular optimization problem subject to a matroid constraint. A centralized low-complexity greedy algorithm is then proposed to obtain a suboptimal cache placement strategy with guaranteed performance. The basic concepts about matroid and submodular function can be found in [26].

A. Matroid Constrained Submodular Optimization

We first define the ground set for cache placement as

$$S = \{f_1^{(1)}, \ldots, f_N^{(1)}, \ldots, f_1^{(M)}, \ldots, f_N^{(M)}\},$$  

(8)

where $f_n^{(m)}$ denotes the event that file $f_n$ is placed in the cache of BS $a_m$. The ground set $S$ contains all possible caching strategies which can be applied in the system. In particular, we use

$$S_m = \{f_1^{(m)}, f_2^{(m)}, \ldots, f_N^{(m)}\} \quad (\forall m = 1, 2, \ldots, M)$$  

(9)

to denote the set of all files that might be placed in the cache of BS $a_m$. Thus, the ground set $S$ can be partitioned into $M$ disjoint sets, i.e., $S = \bigcup_{m=1}^{M} S_m$, $S_m \cap S_{m'} = \emptyset$ for any $m \neq m'$.

Given the finite ground set $S$, we continue to define a partition matroid $M = (S; \mathcal{I})$, where $\mathcal{I} \subseteq 2^S$ is a collection of independent sets defined as:

$$\mathcal{I} = \{X \subseteq S : |X \cap S_m| \leq Q_m, \forall m = 1, 2, \ldots, M\},$$  

(10)

which accounts for the constraint on the cache capacity $Q_m$ at each BS $a_m$ (c.f. (7a)). The set of files placed in the cache of BS $a_m$ can be denoted by $X_m = X \cap S_m$.

Then, we show that the average delay is a monotone supermodular set function over the ground set $S$. Note that every set has an equivalent boolean presentation. For any $X \subseteq S$, the incidence vector of $X$ is denoted by the vector $\mu \in \{0, 1\}^S$ whose $i$-th element is defined as

$$\mu_i = x_{nm}, \quad i = (m-1)N + n,$$  

(11)

where $\mu$ represents the mapping between $X_{nm}$ and $\mu_i$. In the set $X \subseteq S$, $f_n^{(m)} \in X$ indicates $\mu_i = 1$. Otherwise, $\mu_i = 0$. Similarly, the boolean presentation of the subset $X_m$ is denoted by $\mu_m$. In this context, the delay function $\bar{D}_{nk}(X)$ is equivalent to the set function $\bar{D}_{nk}(X)$ over the set $X \subseteq S$. The property of $\bar{D}_{nk}(X)$ is summarized in the following theorem.

**Theorem 2.** $\bar{D}_{nk}(X) = \bar{D}_{nk}(X)$ is a monotone supermodular function defined over $X \subseteq \mathcal{I}$.  

**Proof:** The proof is deferred to Appendix [5].

From [26], the class of submodular functions is closed under non-negative linear combinations. Therefore, for $p_{nk} \geq 0$ with $k = 1, 2, \ldots, K$ and $n = 1, 2, \ldots, N$, the set function

$$\bar{D}(X) = \frac{1}{K} \sum_{k=1}^{K} \sum_{n=1}^{N} p_{nk} \bar{D}_{nk}(X)$$  

(12)

is also monotone submodular.

By taking the partition matroid $M = (S; \mathcal{I})$ (c.f. (10)) into consideration, Problem (7) can be reformulated into a matroid constrained monotone submodular maximization problem:

$$\max_{\mathcal{I}} \bar{D}(X) = \frac{1}{K} \sum_{k=1}^{K} \sum_{n=1}^{N} p_{nk} \bar{D}(X)$$  

(13)

subject to

$$X \in \mathcal{I},$$

where the constraint $X \in \mathcal{I}$ (c.f. (10)) shows that each BS $a_m$ can cache up to $Q_m$ files.
At first, the centralized algorithm can be estimated as

\[ X = \emptyset \text{ and } Y = \emptyset \]

as well as local interactions between BSs in the neighborhood. The basic distributed algorithm for cache placement autonomously, relying on locally collected network-side and user-related information, as well as local interactions between BSs in the neighborhood. In this section, we propose a belief propagation based distributed algorithm to perform cooperative caching. The basic concept of the message passing procedure can be found in Appendix C.

### A. Factor Graph Model for Cache Placement

To apply the belief propagation based distributed algorithm, Problem (7) is first transformed into an unconstrained optimization problem as presented in Lemma 3. To this end, we define two functions of the caching strategy \( X \) as:

\[ \eta_{nk}(X) = \exp \left( -p_{nk} D_{nk}(X) \right), \]

\[ g_m(X) = \begin{cases} 1, & \sum_{n=1}^{N} x_{nm} \leq Q_m, \\ 0, & \text{otherwise}. \end{cases} \]

**Lemma 3.** Let \( C = \{(f_n,u_k) \mid p_{nk} > 0, f_n \in F, u_k \in U\} \) denote the set of all possible pairs of file \( f_n \) and user \( u_k \). Problem (7) is equivalent to the following problem

\[
\hat{X} = \arg \max_{X \in \{0,1\}^{NM}} \prod_{(f_n,u_k) \in C} \eta_{nk}(X) \prod_{m=1}^{M} g_m(X). \tag{18}
\]

**Proof:** Problem (7) is equivalent to maximizing

\[
-\sum_{k=1}^{K} \sum_{n=1}^{N} p_{nk} D_{nk}(X) \]

subject to the constraints

\[
\sum_{n=1}^{N} x_{nm} \leq Q_m, \text{ for all } m.
\]

By introducing the exponential function \( \eta_{nk}(X) \) given by (16) and the indicator function \( g_m(X) \) given by (17), the equivalent optimization problem is converted into a product form, as presented in (18).

In (18), \( \eta_{nk}(X) \) is used to measure the delay performance when transmitting file \( f_n \) to user \( u_k \), and \( g_m(X) \) imposes a strict constraint on the cache capacity of BS \( a_m \).

Then, we present the factor graph model for the optimization problem (18). According to the network topology (e.g., Fig. 2(a)), we introduce a variable node \( \mu_i \) for each element \( x_{nm} \) and a function node \( F_j \) for each function node \( \eta_{nk}(X) \) or \( g_m(X) \), as shown in Fig. 2(b). The mapping rule from \( x_{nm} \) to \( \mu_i \) is given by (11), and the mapping rule from \( \eta_{nk}(X) \) or \( g_m(X) \) to \( F_j \) is expressed as

\[
F_j = \begin{cases} \{\eta_{nk}, \ j = \sum_{l=1}^{K} |F_l| + \xi(n,k), \\ g_m, \ j = \sum_{l=1}^{K} |F_k| + m, \end{cases}
\]

where \( F_k = \{f_n \mid p_{nk} > 0\} \) denotes the set of files which may be requested by user \( u_k \), and \( |F_k| \) is the number of elements in the set \( F_k \), and \( \xi(n,k) \) denotes the index of file \( f_n \) in the set \( F_k \).

In the bipartite factor graph (e.g., Fig. 2(b)), each variable node \( \mu_i \) \( x_{nm} \) is adjacent to the function nodes \( \{F_j\} = \{\eta_{nk}\} \cup \{g_m\} \) for all \( u_k \in U_n \). Similarly, each function node \( F_j = \eta_{nk} \) is connected to the variable nodes \( \{\mu_i = x_{nm}\} \) for all \( a_m \in A_k \). Each function node \( F_j \) \( g_m \) is adjacent to the variable nodes \( \{\mu_i = x_{nm}\} \) for all \( f_n \in F \). Hence, there are \( I = NM \) variable nodes and \( J = M + \sum_{l=1}^{K} |F_k| \) function nodes in this factor graph model.

### B. Message Passing Procedure for Cache Placement

Our goal is to design a message-passing procedure which allows us to gradually approach the optimal solution to (18).
Theorem 4. The message \( \alpha_{i \rightarrow j}^{t+1} \) is updated as

\[
\alpha_{i \rightarrow j}^{t+1} = \sum_{l \in \Gamma^F_j(i)} \beta_{l \rightarrow i}^t.
\]

When \( F_j \neq \eta_{nk} \), the message \( \beta_{j \rightarrow i}^{t+1} \) is given by

\[
\beta_{j \rightarrow i}^{t+1} = p_{nk} \left( \tilde{D}_{nk}(X_{i 0}^t) - \tilde{D}_{nk}(X_{i 1}^t) \right),
\]

where the caching vectors \( X_{i 0}^t \) and \( X_{i 1}^t \) can be obtained by assigning their elements as

\[
x_{nm} \doteq \mu_l = \begin{cases} 1, & l \in E_i \setminus \{i\} \mid \alpha_{i \rightarrow j}^t > 0, \\ 0, & \text{otherwise}, \end{cases}
\]

and

\[
x_{nm} \doteq \mu_l = \begin{cases} 1, & l \in E_i \cup \{i\}, \\ 0, & \text{otherwise}, \end{cases}
\]

respectively. When \( F_j = g_m \), the message \( \beta_{j \rightarrow i}^{t+1} \) is updated as

\[
\beta_{j \rightarrow i}^{t+1} = \min \left\{ 0, -\alpha_{i \rightarrow j}^t(Q_m)(t) \right\},
\]

where \( \alpha_{i \rightarrow j}^t(Q_m)(t) \) is the \( Q_m \)-th message among the messages \( \{\alpha_{i \rightarrow j}^t\} (l \in \Gamma^F_j(i)) \) sorted in the descending order.

Proof: The proof is deferred to Appendix D.

In practice, the messages \( \alpha_{i \rightarrow j}^t \) and \( \beta_{j \rightarrow i}^t \) reflect the beliefs on the value of \( \mu_i \) and should be updated according to (24) and (22) (or (23)), respectively, in each iteration.

2) Belief Update: In the \( t \)-th iteration, the belief on \( \mu_i = x \) is expressed as

\[
b_{i}^{t+1}(x) = \prod_{j \in \Gamma^F_i} m_{F_j \rightarrow \mu_i}^{t}(x),
\]

which is the product of all the messages incident to \( \mu_i \). Hence, the belief ratio in the logarithmic domain can be obtained as

\[
b_i^t = \log \left( \frac{b_i^t(1)}{b_i^t(0)} \right) = \sum_{j \in \Gamma^F_i} \beta_{j \rightarrow i}^t,
\]

where \( \beta_{j \rightarrow i}^t \) is given by (23) for \( F_j = g_m \) and by (22) for \( F_j = \eta_{nk} \) \( j \in \Gamma^F_i \{i\} \), respectively. As a result, the estimation of \( \mu_i \) can be expressed as

\[
\hat{\mu}_i^t = \begin{cases} 1, & \text{if } b_i^t > 0, \\ 0, & \text{if } b_i^t < 0. \end{cases}
\]

In each iteration, each variable node \( \mu_i \) updates its belief on its associated variable \( x_{nm} \) according to (25) and makes an estimate of \( x_{nm} \) according to (26) until it converges.

C. Distributed Cache Placement Algorithm

When we map the message passing procedure derived on the factor graph (e.g., Fig 2(b)) back to the original network graph (e.g., Fig 2(a)), we notice that all the messages are updated at the BSs and some of them be exchanged between neighboring BSs.
Algorithm 2 Distributed algorithm for cache placement

1. Map $\eta_{nk}, g_m$ to $F_j$ and $x_{mn}$ to $\mu_i$ for $\forall n, k, m$.
2. Set $t = 0$ and $\alpha_{i \rightarrow j}^{t} = \beta_{j \rightarrow i}^{t} = 0$, $\forall i, j$.
3. Set $t_{max}$ as a sufficiently large constant.
4. while Not convergent and $t \leq t_{max}$ do
5.   for $m = 1 : M$ do
6.      for $n = 1 : N$ do
7.         Calculate the message $\alpha_{i \rightarrow j}^{t}$ by (21);
8.         for $k \in U_m$ do
9.            Calculate the message $\beta_{j \rightarrow i}^{t}$ for $F_j \triangleq \eta_{nk}$ by (22);
10.        end for
11.       end for
12.      Calculate the message $\beta_{j \rightarrow i}^{t}$ for $F_j \triangleq g_m$ by (23);
13.      Calculate the belief $\hat{b}_i$ by (24);
14.      Estimate each variable $\mu_i$ by (26);
15.    end for
16.   Check the convergence, and set $t = t + 1$;
17. end while
18. Obtain the optimal estimate $\hat{X}$ to the solution of (18).

1) Scenario I: When user $u_k$ is connected to one single BS $a_m$, as shown in Fig.2(b), the update of messages $\alpha_{i \rightarrow j}^{t}$ and $\beta_{j \rightarrow i}^{t}$ is performed at this BS for the variable node $\mu_i \equiv x_{mn}$, the function nodes $F_j \triangleq \eta_{nk}$, and $F_j \triangleq g_m$. In this case, each BS $a_m$ performs the message calculation and belief update for all the users just served by itself, i.e., $u_k \in U_m$ and $|A_k| = 1$.

2) Scenario II: When user $u_k$ is in the coverage of multiple BSs $A_k$, the update of messages $\alpha_{i \rightarrow j}^{t}$ and $\beta_{j \rightarrow i}^{t}$ associated with the function node $F_j \triangleq \eta_{nk}$ is performed at one BS $a_m$ and be exchanged between the serving BSs of this user $A_k$ over control links, as shown in Fig.1.

Notice that message exchanges just take place in Scenario II, and the communication overhead induced depends on the number of common users covered by multiple BSs. From the above discussion, we summarize the message passing based distributed algorithm for cache placement in Algorithm 2. In this algorithm, the message update for each user should be performed just once by one single BS in each iteration. To avoid confusion, $U_m$ is used to denote the set of users whose messages are processed by BS $a_m$ in Algorithm 2.

VI. SIMULATION RESULTS

In this section, we present simulation results to demonstrate the performance of the proposed cache placement algorithms, i.e., Algorithm 1 and Algorithm 2. We consider a Fog-RAN with $M$ BSs and $K$ mobile users. Each BS serves the users in a circular cell with a radius of 150m, and the distance between neighboring BSs is 200m. $K$ users are uniformly and independently distributed in the area covered by the $M$ cells. File requests of each user $u_k$ follow the Zipf distribution with parameter $\gamma_k$. The users in the cell interior are served by just one single BS, while the users in the overlapping area of cells are covered by multiple BSs and thus cooperative transmission may be enabled. The connectivity between the BSs and users is thus established.

Suppose that the system bandwidth is 5MHz, and the length of each time slot is 20ms. The file size is equal to 100Mbits. The path-loss exponent is set as 3.5. The small-scale channel gain $|h_{km}|^2$ follows independently standard exponential distribution in each time slot. Assume that no inter-cell interference is induced by adopting appropriate scheduling policies, and the transmit power is set to make sure that the average received SNR at the cell edge is equal to 0dB. Unless otherwise stated, we set $K = 100$, $M = 10$, and $D_{nk} = 40s$. Suppose that each user $u_k$ requests file $f_n$, with probability $\rho_{nk} = (\phi(n))^{-\gamma_k}$, where $\{\phi(n)\}_{n=1}^{N}$ is a random permutation of $[1, \ldots, N]$, i.e., we assume different users have different request distributions.

In the considered system, we compare two transmission aware caching strategies and two baseline popular caching strategies: 1) Non-cooperative transmission aware caching (Non-CoTC) strategy, which is designed based on prior knowledge that each individual user has the file preference $p_{nk}$ and is served by one serving BS using non-cooperative transmission...
given by (1): 2) Cooperative transmission aware caching (CoTC) strategy, which is designed based on prior knowledge that each individual user has the file preference \( p_{nk} \) and is served by one BS using non-cooperative transmission given by (1), or by multiple BSs using cooperative beamforming given by (2), depending on the connectivity between the user and the BSs; 3) Globally popular caching (GPC) strategy, which caches the most \( Q_m \) popular files at each BS \( a_m \) based on the network-wide file popularity \( \{ p_n \} \). Here, the file popularity is evaluated as \( \bar{p}_n = \frac{1}{N} \sum_{k=1}^{K} p_{nk} \), i.e., the average value of the file preferences of users in the network; 4) Locally popular caching (LPC) strategy, which caches the most \( Q_m \) popular files at each BS \( a_m \) based on the local file popularity \( \bar{p}_m = \frac{1}{|U_m|} \sum_{u_k \in U_m} p_{uk} \), i.e., the average value of the file preferences of users served by the BS \( a_m \). The proposed transmission aware caching strategies can be performed in either a centralized or a distributed way. There is no difference between centralized and distributed ways of performing the Popular caching strategy.

A. Performance Evaluation

We demonstrate the performances of our considered four caching strategies in two scenarios when \( \gamma_k = 0.65 \), \( N = 1000 \) and \( \gamma_k = 0.2 + 4.8 \frac{Q}{N} \), \( N = 200 \) in Fig. 3 and Fig. 4, respectively. In each scenario, we plot the average download delay and hit probability curves of these caching strategies in sub-figures (a) and (b), respectively, for different cache capacities \( Q_m = Q \). When our proposed Non-CoTC or CoTC strategy is applied, the users’ average download delay \( D(X) \) is computed by substituting the solution \( X \) that is achieved either by Algorithm 1 or by Algorithm 2. When the GPC or LPC strategy is applied, the average delay \( D(X) \) is obtained by substituting the GPC or LPC solution \( X \). As shown in Fig. 3 and Fig. 4, the average download delay monotonically decreases with the increase of the cache capacity \( Q \) for any given caching strategy. This is due to the fact that with the increase of storage capacity, more files are cached in each BS and more users can download files from local BSs instead of the content server. Due to the same reason, the users’ average hit probability monotonically increases with the cache capacity.

As shown in Fig. 3(a) and Fig. 4(a), the two transmission aware caching strategies, i.e., Non-CoTC and CoTC, achieve smaller average download delays than the two popular caching strategies, i.e., LPC and GPC, for any cache capacity \( Q \) less than \( N \). Meanwhile, the average hit probabilities of the CoTC and Non-CoTC strategies are higher or equal to that of the LPC strategy, and much higher than the GPC strategies when \( Q < N \), as shown in Fig. 3(b) and Fig. 4(b). This is because the transmission aware caching strategies cache files at the BSs based on the accurate file preferences of individual users and the prior information on content delivery techniques that will be applied by the BSs. While the LPC or GPC strategy performs caching based on the file preference statistics of the users in each cell or in the network, which could not reflect the file preferences of individual users.

The delay performance of the caching strategies not only depends on the users’ hit performance, but also on the transmission schemes the BSs will adopt to deliver the requested files. It is observed from Fig. 3(a) and Fig. 4(a) that the CoTC strategy performs much better than the Non-CoTC strategy in terms of the average delay and hit probability. The delay performance gap between the two transmission aware caching strategies becomes larger as the cache capacity increases, since more files can be cached to facilitate cooperative transmission for cell-edge users. In other words, the CoTC strategy can exploit both caching gain and cooperative gain to reduce the average delay. Hence, the design of caching strategies should not only target at improving the users’ average hit probability, but also bringing more cooperative transmission opportunities. Similarly, the delay performance is significantly improved when cooperative transmission is applied instead of non-cooperative transmission for any caching strategy.

At the same time, the users’ skewness on content popularity has a great impact on the performances of the considered caching strategies. When \( \gamma_k = 0.65 \), each user is interested in a large number of files while only a very small number of files can be cached locally at the serving BSs of each user when \( Q \) is less than \( N \). From Fig. 3(a) the delay gap between the CoTC (or Non-CoTC) strategy and the LPC strategy is not very large. And the GPC strategy which caches the same files in each BS achieves the worst delay and hit performances. When \( \gamma_k = 0.2 + 4.8 \frac{Q}{N} \), the skewness on content popularity is quite different among users. This means some users have interests on many files while some users just have preferences on very few files. In contrast to the case with \( \gamma_k = 0.65 \), a higher proportion of the files that the users may request can be cached at the BSs. Therefore, the delay and hit performances of the considered caching strategies are all improved. And the delay gap between the CoTC (or Non-CoTC) strategy and the LPC strategy becomes very significant especially when the cache capacity \( Q \) is very small. It is also interesting to see that the delay performance of the LPC strategy gets affected by content delivery schemes applied by the BSs. As shown in Fig. 4(a) the LPC strategy always achieves a smaller average delay than the GPC strategy if cooperative transmission is adopted. However, it performs worse in the larger \( Q \) region \((Q > 65)\) when non-cooperative transmission is applied. This happens when some users are served by their serving BSs which have not cached their requested files, since the LPC strategy caches files based on the file preferences of co-located users and pushes quite different contents in each BS.

From Fig. 3 and Fig. 4 the proposed belief propagation based distributed algorithm can achieve a nearly identical delay performance as compared to the centralized greedy algorithm which provides a guaranteed performance \([33]\), i.e., 1/2-approximation in the general case and \((1 - 1/e)\)-approximation in some special cases. It has a slightly larger delay performance in the small-capacity region (e.g., \( Q \) is around 20), and achieves almost the same performance as the centralized algorithm in other scenarios.

B. Approximation of File Preferences

In practice, it is very challenging to accurately estimate the file preference of each individual user due to the lack
of sufficient samples. Instead, each BS may estimate an approximate file preference for all the users in its coverage, i.e., to estimate the average preference. In this part, we discuss the impact of the users’ file request preference statistics, either perfectly or approximately known. In Fig. 5 we show how the average download delay changes with the content popularity skewness. In this experiment, all the users are supposed to have the same preference parameter \( \gamma_k = \gamma \). The cache capacity is set as \( Q = 50 \) and the total number of files is \( N = 100 \). The approximate preference for file \( f_n \) is given by \( \tilde{p}_{nk} = \frac{1}{|U_m|} \sum_{u \in U_m} p_{nk} \) (\( \forall u_k \in U_m \)), i.e., only the statistical average of all the users in the coverage of each BS \( a_m \) is known, while a perfect knowledge \( p_{nk} \) includes preference for each individual user. It is observed that the average delay is significantly reduced when the parameter \( \gamma \) is increased within \( 0.3 \leq \gamma \leq 3 \). In this range, the users have preferences on fewer and fewer files with the increase of the parameter \( \gamma \). This means that more and more requested files are cached at the BSs, and can be transmitted to the users directly. As a result, the average download delay is greatly reduced when \( \gamma \) is increased within \( 0.3 \leq \gamma \leq 3 \). When \( \gamma > 3 \), almost all the requested files have been cached and the average download delay is nearly equal to the average transmission time from the BSs to the users. In this case, the change of the average delay is not obvious. In Fig. 5 we also plot the average delay performance when approximate file preferences instead of accurate file preferences are applied. It can be seen that the delay gap is very small.

In Fig. 6 we plot the iterative procedure of the belief propagation based distributed algorithm for different storage capacities \( Q_m \) and \( N = 100 \). In this experiment, the CoTC strategy is performed in a distributed way. It is observed that the average delay starts from an initial value, fluctuates up to dozens of iterations and gradually converges to a suboptimal solution.
veloped transmission aware cache placement strategies in both Fog-RANs, by taking into account different file preferences. Each BS run computations and exchange very few messages to place files at the BSs based on locally collected information. A belief propagation based distributed algorithm was proposed since more elements are added greedily and more iterations are required. In contrast, the centralized algorithm increases very slowly with the increase of the cache capacity. This indicates that the cache capacity has a greater impact on the computation complexity of the centralized algorithm rather than the distributed algorithm, since more elements are added greedily and more iterations are processed in the centralized algorithm when the cache capacity is increased. When applying the distributed algorithm, the cache capacity is a parameter which only adjusts the value of the messages during iterations. It does not change the factor graph model, and hence may not cause a significant impact on its computation complexity.

**C. Algorithm Complexity**

We now discuss the computation complexity of our proposed centralized and distributed algorithms when performing the CoTC strategy. Here, we measure the computation complexity by the number of calculations required in the algorithms. In Fig. 7 we plot the computation complexity of the proposed algorithms versus the cache capacity $Q$. In this experiment, the number of BSs and the number of users are set as $M=10$ and $K=100$, and the total number of files is set to be 100. It can be seen that the computation complexity of the centralized algorithm rapidly increases with the increase of the cache capacity $Q$, while the computation complexity of the distributed algorithm increases very slowly with the cache capacity $Q$. This indicates that the cache capacity has a greater impact on the computation complexity of the centralized algorithm rather than the distributed algorithm, since more elements are added greedily and more iterations are required in the centralized algorithm when the cache capacity $Q$ is increased. When applying the distributed algorithm, the cache capacity is a parameter which only adjusts the value of the messages during iterations. It does not change the factor graph model, and hence may not cause a significant impact on its computation complexity.

**VII. CONCLUSIONS**

In this work, we studied the cache placement problem in Fog-RANs, by taking into account different file preferences and diverse transmission opportunities for each user. We developed transmission aware cache placement strategies in both centralized and distributed operation modes of Fog-RANs. In the centralized mode, a low-complexity centralized greedy algorithm was proposed to achieve a suboptimal solution within a constant factor to the optimum using submodular optimization techniques. In the distributed mode, a low-complexity belief propagation based distributed algorithm was proposed to place files at the BSs based on locally collected information. Each BS run computations and exchange very few messages with its neighboring BSs iteratively until convergence. By simulations, we showed that both of the proposed algorithms can not only improve the users’ cache hit probability but also provide more flexible cooperative transmission opportunities for the users. As a result, our proposed centralized and distributed cache placement algorithms can significantly improve the file delivery performance by providing cooperative transmission opportunities for mobile users to the maximum extent. It was also shown that the distributed cache placement algorithm can achieve an average delay performance comparable to the centralized cache placement algorithm while spending much less calculations in each individual BS.

**APPENDIX**

**A. Proof of Theorem 7**

In the scenario when file $f_n$ has been cached in one or multiple serving BSs of user $u_k$, i.e., $\sum_{n_m \in A_k} x_{nm} \neq 0$, the associated BSs can transmit to user $u_k$ with rate $R_{nk}(X, i)$ (c.f. (1)-(2) by applying some specific transmission scheme. Since the channel coefficients $h_{km}(i)$ are i.i.d. across the time slots $\{i\}$, the file delivery rates $R_{nk}(X, i)$ are i.i.d. random variables. Hence, the stopping time of completing the transmission of file $f_n$, $T_{nk}(X)$ given by (3), is also a random variable. Based on the definition of channel capacity, we have $R_{nk}(X, i) \geq 0$ for $i = 1, 2, \cdots, T_{nk}(X)$. According to Wald’s Equation in martingale theory [3], we have

$$E_h \left\{ \sum_{i=1}^{T_{nk}(X)} R_{nk}(X, i) \right\} = E_h \left\{ R_{nk}(X) \right\} = \frac{|f_n|}{\Delta t}.$$  

Therefore, the average download delay is expressed as

$$D_{nk}(X) = E_h \left\{ T_{nk}(X) \cdot \Delta t \right\} = \frac{|f_n|}{E_h \left\{ R_{nk}(X) \right\}}.$$  

when file $f_n$ is cached in the associated BSs with $\sum_{n_m \in A_k} x_{nm} \neq 0$. When $\sum_{n_m \in A_k} x_{nm} = 0$, file $f_n$ has not been cached in any serving BS of user $u_k$. The BSs $A_k$ download this file from the content server by the backhaul link and then transmit to user $u_k$ over the wireless channel. Accordingly, the average delay can be estimated by $D_{nk}(X) + \frac{|f_n|}{E[R_{nk}(X_{k})]}$ where $D_{nk}(X)$ is the extra delay of file delivery from the content server to the serving BSs $A_k$, and $R_{nk}(X_k)$ is the rate at which the BSs $A_k$ transmit file $f_n$ to user $u_k$ over wireless channel. Here, $X_k$ is an equivalent caching strategy indicating that file $f_n$ can be downloaded from the BSs $A_k$ by user $u_k$. Thus, the average delay $D_{nk}(X)$ is established in (5).

**B. Proof of Theorem 4**

From Theorem 4 we have the average delay of downloading file $f_n$ for user $u_k$ presented in (5) can also be expressed as

$$D_{nk}(X) = \begin{cases} \frac{|f_n|}{R_{nk}(X)} \cdot \min_{m \in A_k} x_{nm}, & \text{if } \sum_{m=1}^{M} x_{nm} \neq 0, \\ \frac{|f_n|}{\sum_{m=1}^{M} x_{nm}}, & \text{otherwise}, \end{cases}$$

where $R_{nk}(X) = \mathbb{E} \{ B \log(1 + Y_{nk}(X)) \}$ with $Y_{nk}(X) = \sum_{m=1}^{M} |h_{km}|^2 x_{nm} \text{SINR}_m$ representing the received SINR.
We will show that the average delay \( \bar{D}_{nk}(X) = -\bar{D}_{nk}(X) \) is a monotone submodular function.

Let \( X \subseteq X' \subseteq I \), and \( s \in S \cap X' \). The incidence vectors for \( X \) and \( X' \) are denoted by \( X = [x_{nm}] \) and \( X' = [x_{nm}] \), respectively. If \( s \neq f_{nm}^\dagger \) for any \( m \in A_k \), we have \( \bar{D}_{nk}(X \cup \{s\}) - D_{nk}(X) = \bar{D}_{nk}(X' \cup \{s\}) - D_{nk}(X') = 0 \). We then consider the case when \( s = f_{nm}^\dagger \) for any \( m \in A_k \).

**Case I:** \( X \subseteq X' \subseteq I \) and \( s \neq 0 \). The following inequality holds:
\[
\bar{D}_{nk}(X' \cup \{s\}) - D_{nk}(X') < 0,
\]
where the inequality (a) comes from \( \bar{D}_{nk}(X \cup \{s\}) < \bar{D}_{nk}(X') \). The gap between \( \bar{D}_{nk}(X' \cup \{s\}) \) and \( D_{nk}(X') \) is equal to
\[
\sum_{m \in A} x_{nm} \geq \sum_{m \in A_k} x_{nm}.
\]

**Case II:** \( X \subseteq X' \subseteq I \) and \( s \neq 0 \). We have
\[
\bar{D}_{nk}(X \cup \{s\}) - D_{nk}(X) = \bar{D}_{nk}(X' \cup \{s\}) - D_{nk}(X') = 0.
\]

**Case III:** \( X \subseteq X' \subseteq I \) and \( s \neq 0 \). We have
\[
\bar{D}_{nk}(X' \cup \{s\}) - D_{nk}(X') = 0.
\]

According to the definition of \( \bar{D}_{nk}(X) \), we have \( \bar{D}_{nk}(X' \cup \{s\}) = \bar{D}_{nk}(X') \), and the inequality (b) holds.

The following inequality hold:
\[
\bar{D}_{nk}(X \cup \{s\}) = \bar{D}_{nk}(X'),
\]
where the inequality (a) comes from \( \bar{D}_{nk}(X' \cup \{s\}) \leq \bar{D}_{nk}(X') \) and \( \bar{D}_{nk}(X \cup \{s\}) \leq \bar{D}_{nk}(X') \) for any \( m \in A_k \).

**Case IV:** \( X \subseteq X' \subseteq I \) and \( s \neq 0 \). We have
\[
\bar{D}_{nk}(X' \cup \{s\}) - D_{nk}(X') = 0.
\]

According to the definition of \( \bar{D}_{nk}(X) \), we have \( \bar{D}_{nk}(X' \cup \{s\}) = \bar{D}_{nk}(X') \), and the inequality (b) holds.

The following inequality hold:
\[
\bar{D}_{nk}(X \cup \{s\}) = \bar{D}_{nk}(X'),
\]
where the inequality (a) comes from \( \bar{D}_{nk}(X' \cup \{s\}) \leq \bar{D}_{nk}(X') \) and \( \bar{D}_{nk}(X \cup \{s\}) \leq \bar{D}_{nk}(X') \) for any \( m \in A_k \).

**Case V:** \( X \subseteq X' \subseteq I \) and \( s \neq 0 \). We have
\[
\bar{D}_{nk}(X' \cup \{s\}) - D_{nk}(X') = 0.
\]
\[ m_{F_j \rightarrow \mu_i}(0) = \max_{E_i^j} \left\{ \exp(-p_{nk}D_{nk}(X^{(0)})) \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(1) \right\} \times \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(0) \]  

\[ \beta_{j ightarrow 1}^{t+1} = \max_{E_i^j} \left\{ (-p_{nk}D_{nk}(X^{(1)})) + \sum_{i \in E_i^j} \alpha_{i \rightarrow j} \right\} \right\} \times \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(0) \]  

(34)

where \( E_i^2 \subseteq \Gamma_i^F \setminus \{i\} \) is also a subset of the index set \( \Gamma_i^F \setminus \{i\} \) such that its associated elements in \( X^{(0)} \) are equal to one, while the other elements are zero with \( \mu_l = 0 \) for all \( l \in \Gamma_i^F \setminus E_i^2 \). From (33) and (34), the message \( \beta_{j ightarrow 1}^{t+1} \) can be expressed as

\[ \beta_{j ightarrow 1}^{t+1} = \max_{E_i^j} \left\{ (-p_{nk}D_{nk}(X^{(1)})) + \sum_{i \in E_i^j} \alpha_{i \rightarrow j} \right\} \right\} \times \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(0) \]  

(35)

where \( X^{(0)} \) and \( X^{(1)} \) are set as caching vectors by selecting the variable nodes \( \{\mu_l\} \) with positive \( \alpha_{i \rightarrow j} \), i.e., \( l \in E_i^+ = \{i \in \Gamma_i^F \setminus \{i\} | \alpha_{i \rightarrow j} > 0\} \), and assigning their associated elements to one. Thus, we have \( \mu_l = x_{lm} = 1 \) for all \( l \in E_i^+ \) in \( X^{(0)} \) and \( \mu_l = x_{lm} = 1 \) for all \( l \in E_i^+ \setminus \{i\} \) in \( X^{(1)} \). This means that each function node \( F_j \) should select its neighboring variable nodes \( \mu_l \) with positive input message \( \alpha_{i \rightarrow j} \) and then calculate the delay gap between \( D_{nk}(X^{(0)}) \) and \( D_{nk}(X^{(1)}) \).

**Case II:** Derivation of \( \beta_{j ightarrow 1}^{t+1} \) for \( F_j \neq g_m \). By substituting the constraint function into (32), the message \( m_{F_j \rightarrow \mu_i}(1) \) when \( F_j \neq g_m \) can be represented as

\[ m_{F_j \rightarrow \mu_i}(1) = \max_{E_i^j} \left\{ g_m(X^{(1)}) \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(1) \right\} \times \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(0) \]  

(36)

where \( E_i^3 \) is a subset of the index set \( \Gamma_i^F \setminus \{i\} \) and \( |E_i^3| \leq Q_m - 1 \). This means that to satisfy the cache capacity constraint, there exist at most \( Q_m - 1 \) neighboring variable nodes \( \{\mu_l\} \) with \( \mu_l = 1 \) (in \( E_i^3 \)) except the variable node \( \mu_i = 1 \). Similarly, we can compute the message \( m_{F_j \rightarrow \mu_i}(0) \) when \( F_j \neq g_m \) as

\[ m_{F_j \rightarrow \mu_i}(0) = \max_{E_i^j} \left\{ g_m(x^{(0)}) \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(1) \right\} \times \prod_{i \in E_i^j} m_{\mu_i \rightarrow F_j}(0) \]  

(37)

where \( E_i^4 \) is a subset of the index set \( \Gamma_i^F \setminus \{i\} \) and \( |E_i^4| \leq Q_m - 1 \). Since \( \mu_l = 0 \), there exist at most \( Q_m \) neighboring variable nodes \( \{\mu_l\} \) (in \( E_i^4 \)) with \( \mu_l = 1 \) to satisfy the cache capacity constraint. From (36) and (37), the message ratio of \( m_{F_j \rightarrow \mu_i}(1) \) and \( m_{F_j \rightarrow \mu_i}(0) \) in the logarithmic domain can be expressed as

\[ \beta_{j ightarrow 1}^{t+1} = \max_{E_i^j} \left\{ \sum_{i \in E_i^2} \alpha_{i \rightarrow j} \right\} - \max_{E_i^j} \left\{ \sum_{i \in E_i^3} \alpha_{i \rightarrow j} \right\} \]  

(38)

By sorting the messages \( \{\alpha_{i \rightarrow j}\} \) \((\forall l \in \Gamma_i^F \setminus \{i\})\) in the decreasing order as \( \alpha_{i_1 \rightarrow j}, \alpha_{i_2 \rightarrow j}, \cdots, \alpha_{i_{Q_m-1} \rightarrow j} \), ..., we can further simplify \( \beta_{j ightarrow 1}^{t+1} \) as

\[ \beta_{j ightarrow 1}^{t+1} = \begin{cases} \min\{0, -\alpha_{i_{Q_m-1} \rightarrow j}\} & \text{if } \alpha_{i_{Q_m-1} \rightarrow j} \geq 0, \\ 0 & \text{otherwise,} \end{cases} \]  

(39)

which is exactly equal to \( \min\{0, -\alpha_{i_{Q_m-1} \rightarrow j}\} \), as given by (38).
[21] A. Abboud, E. Baştuğ, K. Hamidouche, and M. Debbah, “Distributed caching in 5G networks: An alternating direction method of multipliers approach,” in Proc. IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Stockholm, Sweden, June 28–July 1 2015.

[22] R. Wang, X. Peng, J. Zhang, and K. B. Letaief, “Mobility-aware caching for content-centric wireless networks: Modeling and methodology,” IEEE Commun. Mag., vol. 54, no. 8, pp. 77–83, Aug. 2016.

[23] K. Poularakis, G. Iosifidis, and L. Tassiulas, “Approximation algorithms for mobile data caching in small cell networks,” IEEE Trans. Commun., vol. 62, no. 10, pp. 3665–3677, Oct. 2014.

[24] M. Dehghan, A. Seetharam, B. Jiang, T. He, T. Salonidis, J. Kurose, D. Towsley, and R. Sitanaman, “On the complexity of optimal routing and content caching in heterogeneous networks,” in Proc. INFOCOM, April 26–May 1 2015.

[25] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and Zipf-like distributions: Evidence and implications,” in Proc. IEEE INFOCOM, Mar. 1999, pp. 126–134.

[26] A. Schrijver, Combinatorial optimization: Polyhedra and efficiency. Berlin: Springer, 2003.

[27] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, “Factor graphs and the sum-product algorithm,” IEEE Trans. Inf. Theory, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[28] E. Baştuğ, M. Bennis, and M. Debbah, “A transfer learning approach for cache-enabled wireless networks,” in Proc. 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), May 2015, pp. 161–166.

[29] B. N. Bharath, K. G. Nagananda, and H. V. Poor, “A learning-based approach to caching in heterogeneous small cell networks,” Aug. 2015. [Online]. Available: [http://arxiv.org/abs/1508.03517](http://arxiv.org/abs/1508.03517)

[30] R. Mudumbai, G. Barriac, and U. Madhow, “On the feasibility of distributed beamforming in wireless networks,” IEEE Trans. Wireless Commun., vol. 6, no. 5, pp. 1754–1763, May 2007.

[31] David Williams, Probability with Martingales. Cambridge University Press, 1991.

[32] 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, Dec. 2013.

[33] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, “Cost-effective outbreak detection in networks,” in Proc. 13th ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD), 2007, pp. 420–429.