sensors

Article

Explicit Content Caching at Mobile Edge Networks with Cross-Layer Sensing

Lingyu Chen [1,*], Youxing Su [1], Wenbin Luo [1], Xuemin Hong [1,2] and Jianghong Shi [1,2]

1 School of Information Science and Technology, Xiamen University, Xiamen 361005, China; 23320161153391@stu.xmu.edu.cn (Y.S.); 23320151154076@stu.xmu.edu.cn (W.L.); xuemin.hong@xmu.edu.cn (X.H.); shijh@xmu.edu.cn (J.S.)

2 Key Lab of Underwater Acoustic Communication and Marine Information Technology, Ministry of Education, Xiamen 361005, China

* Correspondence: chenly@xmu.edu.cn; Tel.: +86-592-258-0150

Received: 16 February 2018; Accepted: 18 March 2018; Published: 22 March 2018

Abstract: The deployment density and computational power of small base stations (BSs) are expected to increase significantly in the next generation mobile communication networks. These BSs form the mobile edge network, which is a pervasive and distributed infrastructure that can empower a variety of edge/fog computing applications. This paper proposes a novel edge-computing application called explicit caching, which stores selective contents at BSs and exposes such contents to local users for interactive browsing and download. We formulate the explicit caching problem as a joint content recommendation, caching, and delivery problem, which aims to maximize the expected user quality-of-experience (QoE) with varying degrees of cross-layer sensing capability. Optimal and effective heuristic algorithms are presented to solve the problem. The theoretical performance bounds of the explicit caching system are derived in simplified scenarios. The impacts of cache storage space, BS backhaul capacity, cross-layer information, and user mobility on the system performance are simulated and discussed in realistic scenarios. Results suggest that, compared with conventional implicit caching schemes, explicit caching can better exploit the mobile edge network infrastructure for personalized content dissemination.

Keywords: edge computing; caching; content delivery; content recommendation; cross-layer design

1. Introduction

The mobile Internet is facing great challenges in coping with ever increasing traffic demand. According to a Cisco white paper, the total amount of global mobile data traffic will reach 49 exabytes per month by 2021 [1]. To cope with the exploding traffic demand, short-range and low-cost small base stations (BSs) will be densely deployed to scale up the capacity of the mobile communication network [2–5]. These BSs form the mobile edge network, which is a pervasive and distributed infrastructure. It is further envisioned that the BSs will be equipped with extra computing, sensing and caching resources [6], thereby transforming the mobile edge network from a communications-specific infrastructure to a general-purpose edge computing infrastructure [7–9]. It is anticipated that such a transform can empower a new wave of location-based and time-sensitive applications [10].

Traffic measurements revealed that the dominant application of the mobile communication network (in terms of the percentage of consumed traffic) has shifted from connection-oriented services, such as phone calls and text messages, to content retrieval services, such as web browsing and video streaming [11]. This fact has inspired the research field of content-centric mobile networks (CCMNs). CCMN recognized that content data/traffic has different characteristics compared with real-time communication data/traffic. For example, content data can be cached in multiple locations before user requests [12], users' personal preferences for content can be predicted with fair accuracy [13], and the content popularity follows a long-tail distribution [14]. By carefully leveraging these characteristics, various CCMN designs have been proposed to optimize the mobile network for massive content delivery [15]. Among these designs, distributed caching at mobile BSs has emerged as a promising solution that can exploit the pervasive BS infrastructure for effective delivery of massive content data.

BS caching designs can be broadly categorized into two types: implicit caching and explicit caching. Implicit caching means that the cached content is transparent to users, i.e., users are unaware of what contents are cached at the BSs [16]. In this case, users' requested content is unconstrained and can be any piece of data available on the Internet. a request is answered locally if the target content is cached at the BS (which is called a 'hit'), otherwise the request is served by a remote server [17,18]. The aim of implicit caching is to cache a subset of contents to maximize the overall possibility of getting a 'hit'. Implicit caching is a matured technology in the wired Internet [19], but only recently did its application in mobile BSs attract significant research interest. Ref. [20] proposed a content updating method in cache helpers, taking into account the constraint of backhaul capacity and time-varying contents popularity. Ref. [21] conducted a systematic study by exploiting the user mobility in cache-enabled content-centric wireless networks. Ref. [22] investigated the distributed content placement and delivery schemes based on the Manhattan mobility model. Ref. [23,24] exploited the higher-layer knowledge of user mobility and data request preference to pro-actively cache data and provide seamless handover. In summary, most BS caching schemes proposed to date fall into the category of implicit caching. Various aspects such as user demand prediction [25], backhaul constraint [26], and user mobility [27] have been studied in depth. The main advantage of implicit caching lies in decoupling content request and content delivery, therefore the cache facility becomes a communication infrastructure that can provide added value to a broad range of content providers. However, the effectiveness of implicit caching diminishes with a reducing number of users sharing the same cache [28] because the probability of getting a "hit" reduces. Generally speaking, implicit caching would be more effective when deployed closer to the core network at networks that aggregate a large volume of traffic from many users. As cellular networks evolve toward dense deployment of small cells with fewer users per cell [29], the deployment of implicit caching at the mobile edge network is not cost-effective.

To overcome the drawback of implicit caching, we recently proposed the explicit caching scheme [30,31] as an alternative caching paradigm that can better exploit the increasingly dense small cell infrastructure. It relies on the edge computing capability to stored selective content at a BS and exposed the cached content to local users for interactive, localized browsing and download. In this case, users' requested content is constrained to what is already cached locally. Unlike implicit caching, explicit caching is a different type of edge computing service, which offers joint content recommendation and delivery service to end users. The fundamental advantage of explicit caching is that its utility value (i.e., effectiveness) scales linearly with BS density and cell capacity. In contrast, we note that the effectiveness of the conventional implicit caching scheme decreases at higher BS density. Nevertheless, the explicit caching system also brings unique challenges. The first challenge is to ensure the relevance and attractiveness of cached content. The second challenge is to guarantee that the locally-generated traffic does not degrade the incumbent traffic coming to/from the core network. The third challenge is to guarantee the reliability and timeliness of local content delivery to sustain a satisfactory quality-of-experience (QoE). These three challenges urge a holistic cross-layer design that can jointly optimize content recommendation and content delivery in the explicit caching system.

To the best of our knowledge, research about explicit BS caching systems is still in its infancy. In a conference version of this work [31], we made an initial attempt to propose a cross-layer design architecture and present some initial results. This article extends [31] to a systematic study. Our main contributions are summarized below:

- First, a comprehensive and extended system model is presented, taking into account various aspects such as content property, user interest, user mobility, BS cache space, BS backhaul capacity, and cross-layer sensing capability.
- Second, theoretical performance bounds are derived to estimate the utility of the explicit caching system in a simple yet representative scenario.
- Third, optimal algorithms are given solving the explicit caching problem. Low-complexity heuristic algorithms are also proposed and showed to yield marginal performance degradation compared with the optimal ones.
- The impacts of various parameters on the system performance are systematically evaluated and discussed via simulations. Key design guidelines of the explicit caching system are summarized for future system developers.

The remainder of this paper is organized as follows. Section 2 introduces the system model. Section 3 formulates the content placement problem with cross-layer considerations. Section 4 presents the optimal and heuristic content caching algorithms. Section 5 analyzes the theoretical performance bounds of the system with a simplified model, followed by simulation-based performance evaluation with realistic models in Section 6. Finally, Section 7 concludes the paper.

2. System Model

2.1. Cellular Network Model and Description of the Explicit Caching Service

As illustrated in Figure 1, we consider a mobile edge network with multiple BSs and users, where each user is associated with only one BS. Each BS runs independently to select and retrieve content from a remote Internet Data Center (IDC) and stores the content at a local cache. The explicit caching service allows a user to browse the cached content of the associated BS and download/view the content according to their personal interest. Essentially, the explicit caching service resembles an FTP service in a local network. The backhaul link connecting each BS and the IDC can either be fixed or wireless connections, while each link is characterized by a limited capacity. Time is slotted into periods of T_s seconds. At the beginning of each slot, a BS makes a decision on how to update its cached content. The decision should simultaneously address multiple concerns including the content's attractiveness to local users (i.e., the recommendation problem), the constraints of cache space and backhaul capacity (i.e., the caching problem), and the constraint of radio access capacity (i.e., the content delivery problem). It is assumed that the BSs run distributed and independent algorithms of explicit caching, hence our study can focus on a single typical BS.

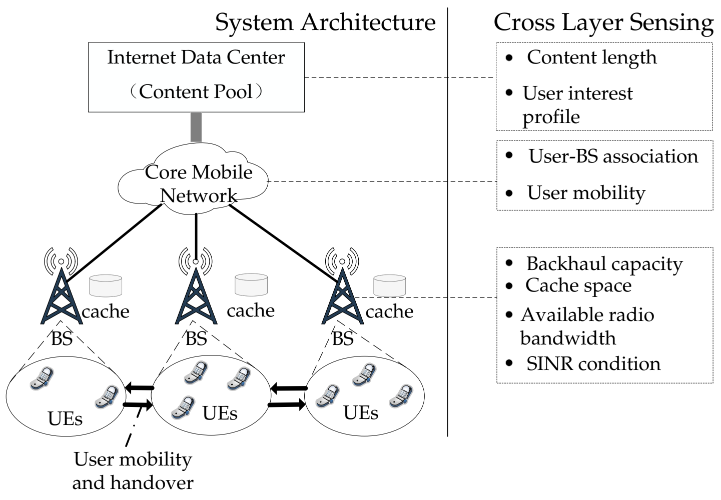

Figure 1. Illustration of the system model.

From the mobile operators' perspective, it is very important for the explicit caching service to coexist peacefully with other existing cellular services. Following the proposal in [30], we assume that the explicit caching service is offered as a "secondary service", which means that it has a lower priority to utilize the radio resource at the BS than conventional services (which are primary services). After radio resource contention, we assume that only a portion of the system bandwidth is left for the explicit caching service. The available bandwidth is assumed to be stable for the time frame of interest and is denoted as W. Apart from the radio bandwidth, other physical resource constraints include the cache storage space S (bits) and backhaul capacity δ (bits). The backhaul capacity caps the maximum number of bits updated between two consecutive time slots.

Users in the cellular network are assumed to be mobile. Let $\mathbb{K}$ denote the set of potential users that may be served by the BS over the long term. Users are assumed to be mobile, which means that they move in and out of coverage from time to time. User mobility is captured by a two-state Markov chain, whose transition probability matrix P_0 is given by

$$\mathbf{P}_0 = \begin{bmatrix} 1-P_a & P_a \\ P_b & 1-P_b \end{bmatrix}, \tag{1}$$

where P_a and P_b denote the probability for a user to move within and out of coverage, respectively. At a time instance, only a subset of users are connected to the BS. The set of connected users is denoted as $\mathbb{U}$. The sizes of $\mathbb{K}$ and $\mathbb{U}$ are denoted as K and U, which represent the number of all users and the number of connected users in a cell, respectively.

2.2. Content Models

It is assumed that a large content pool $\mathbb{F}$ with F pieces of files are stored at the remote IDC. The length of the f-th $(f = 1, ..., F)$ file is denoted by L_f. Based on users' preference, a recommender system computes a numerical value $I_{k,f}$ to indicate the interest of the k-th $(k = 1, 2, ..., K)$ user on the f-th file. The user interest model is captured by a $K \times F$ matrix $\mathbf{I}$, whose entries are given by $I_{k,f}$. For an arbitrary pair of k and f, we assume that $I_{k,f}$ follows independent and identical distributions (i.i.d.). Similarly, L_f also follows i.i.d. distributions. Different models can be used to describe the statistical properties of $I_{k,f}$ and L_f. In our paper, we further distinguish two content models: a simple model and a realistic model. The former is used to facilitate theoretical analysis, while the latter is used for simulations:

- Simple content model: The user interest parameter $I_{k,f}$ is assumed to be a binary random variable $\in \{0, 1\}$ with mean ε. The file length L_f follows a uniform distribution in $[l_{min}, l_{max}]$.
- Realistic content model: According to reported measurements, we assume that $I_{k,f}$ is jointly characterized by two distributions. First, the average popularity of different files (i.e., $\sum_{k=1}^{K} I_{k,f}/K$) follows a Zipf distribution with parameter β [14]; second, the interests of different users on a particular file follow a Gaussian distribution [32]. The file length L_f follows a log-normal distribution [33].

2.3. Transmission Model

Let $D_{k,f}$ denote the time required for the k-th user to download the f-th file. We have

$$D_{k,f} = L_f / C_k, \tag{2}$$

where L_f is the file length and C_k is the instantaneous downlink capacity of the k-th user. The file length L_f is a random variable following a log-normal distribution [33], while the capacity is given by

$$C_k = \frac{W}{U} \log_2 \left(1 + \frac{P_t d_k^{-\alpha} g_m}{W\sigma^2/U + I_k} \right), \tag{3}$$

where P_t is the transmit power per user, σ^2 is the constant noise power of each user, and I_k denotes the accumulated interference perceived by the k-th user. In practice, I_k is dominated by the accumulated inter-cell interference from other co-channel cells. According to the literature of interference modeling and capacity analysis in large scale cellular networks, the accumulated inter-cell interference can be approximately treated as Gaussian noise when evaluating the link capacity [34]. This widely used approximation is also adopted in our paper for simplicity. d_k is the distance between BS and the k-th user, α is the path-loss exponent, and g_m is a fast fading coefficient following an exponential distribution of unit mean (i.e., Rayleigh fading). It is assumed that the available bandwidth W is equally shared among users. Let γ_k denote the k-th user's SINR (Signal to Interference plus Noise Ratio) averaged over small scale fading, we can then rewrite Equation (3) as

$$C_k = \frac{W}{U}\log_2\left(1+\gamma_k g_m\right). \tag{4}$$

2.4. User QoE Model

The ultimate goal of the explicit caching system is to maximize the users' QoE of content consumption. Two factors are considered to have critical impacts on users' QoE. The first factor is the attractiveness of content, which is captured by the user interest parameter $I_{k,f}$. The second factor is the time/delay $D_{k,f}$ for downloading a content file. Each user is assumed to have the same delay tolerance, which is captured by a time threshold T_0. The QoE for the k-th user to download and view the f-th content is then defined as

$$Q_{k,f} = I_{k,f} Z_{k,f}, \tag{5}$$

where $Z_{k,f}$ is defined as

$$Z_{k,f} = \begin{cases} \xi, & D_{k,f} \le T_0, \\ 0, & D_{k,f} > T_0. \end{cases} \tag{6}$$

In Equation (6), ξ is a scaling constant with a positive real value. The above definition means that, if a requested file can be delivered in time, the QoE has a positive value proportional to the user' interest on the file; otherwise, the QoE is zero.

Finally, for the convenience of readers, Table 1 summarizes the major parameters and their physical meanings in our system model.

Table 1. Major system parameters and their physical meanings.

System Symbols	Physical Meaning of Symbols	System Symbols	Physical Meaning of Symbols
$\mathbb{K}$	The set of users that may be served by the BS over a long term	K	The size of $\mathbb{K}$
$\mathbb{U}$	The set of connected users to the BS	U	The size of $\mathbb{U}$
$\mathbb{F}$	The large content pool	F	The size of $\mathbb{F}$
$D_{k,f}$	The time required for the k-th user to download the f-th file	$Q_{k,f}$	The QoE for the k-th user to download and view the f-th content
$I_{k,f}$	The interest of the k-th user on the f-th file	$Z_{k,f}$	Delivery evaluation for the k-th user to download the f-th content
L_f	The file length	C_k	Instantaneous downlink capacity of the k-th user
T_s	Duration of one time slot	T_0	Time threshold of effective distribution
W	Radio bandwidth	P_t	Transmit power per user
d_k	Distance between BS and the k-th user	α	The path-loss exponent
g_m	Fast fading coefficient	σ^2	The power spectrum density of sum noise and interference in the cell
γ_k	The k-th User's SINR averaged over small scale fading	ξ	Scaling constant representing the QoE of an effective distribution
S	Cache storage space of the BS	δ	Backhaul capacity of the BS

3. Problem Formulation with Cross-Layer Consideration

3.1. General Problem Formulation

The aim of this paper is to find a content placement policy that can maximize the expected QoE of users over a long time. The update of content is subject to constraints on the cache space S and backhaul capacity δ. The cache space constraint is memoryless in a sense that it does not impose connections between two consecutive time slots.

However, as illustrated in Figure 2, the backhaul capacity constraint introduce historical dependencies into the system and desires joint considerations over multiple time slots. As a result, our paper will address the general problem of content placement over multiple time slots, noting that the single-slot problem is a special case of the general problem. We further note that the multi-slot problem implies that the information of critical system status (e.g., user association) is available for all the slots under consideration. In practice, this can be achieved by forward prediction (assuming ideal prediction accuracy).

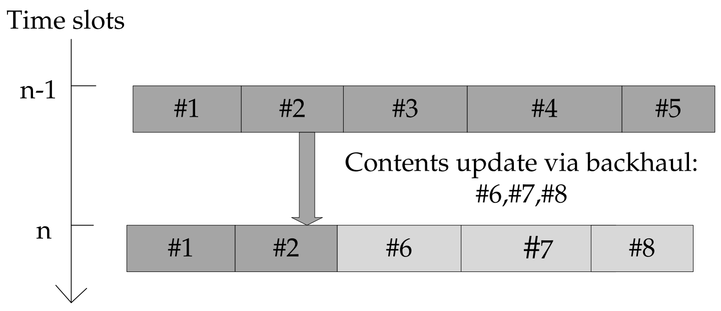

Figure 2. Illustration of content update between two successive time slots, (#6, #7, #8) represent the contents updated via the backhaul network.

Let n denote the index of time slot. Further define a binary variable $x_f(n) \in \{0, 1\}$, which denotes whether the f-th file is cached (taking value 1) or not (taking value 0) during the n-th time slot. The required backhaul traffic for a transition from the $(n-1)$-th slot to the n-th slot can be evaluated as

$$\sum_{f=1}^{F} L_f \{x_f(n) - x_f(n-1)\} x_f(n). \tag{7}$$

Let $V_f(n)$ denote the expected QoE per user for the f-th file during the n-th time slot. We have

$$\begin{aligned} V_f(n) &= \mathbf{E}\left\{ \frac{1}{U(n)} \sum_{k \in \mathbb{U}} Q_{k,f}(n) \right\} \\ &= \frac{1}{U(n)} \sum_{k \in \mathbb{U}} I_{k,f} \mathbf{E}\{Z_{k,f}(n)\}, \end{aligned} \tag{8}$$

where $\mathbf{E}\{\cdot\}$ means taking expectation. Here, the expectation is taken over the fast channel fading coefficient. The users' interest profile is assumed to be consistent over multiple time slots. However, the number of users connected to the BS, as well as the user-BS distances, change from slot to slot. Therefore, $U(n)$, $C_k(n)$ and $Z_{k,f}(n)$ should all be treated as time-varying functions. According to Equation (8), the value of a file/content is measured by the sum QoE of all the active users of a BS. a user is active when he is currently associated with the BS and runs the explicit caching service. Fairness among active users is embedded in such a QoE measure because each active user essentially "votes" to cache the most interested and deliverable contents. In practice, the set of active users changes dynamically and an inactive user may become active. In this case, our caching policy can dynamically update the cached content via backhaul to ensure user fairness among the new set of active users.

The problem of joint content placement and delivery, which aims to maximize the expected user QoE over a time span of N slots, can then be formulated as

$$
\begin{aligned}
(P_1) \quad & \max_{x} \sum_{n=1}^{N} \sum_{f=1}^{F} V_f(n) x_f(n) \\
& st. \sum_{f=1}^{F} L_f x_f(n) \leq S, \forall n \\
& \sum_{f=1}^{F} L_f (x_f(n) - x_f(n-1)) x_f(n) \leq \delta, \forall n \\
& x_f(n) \in \{0,1\}.
\end{aligned} \tag{9}
$$

The optimization problem in (P_1) formulates the explicit caching problem as a joint content recommendation, caching, and delivery problem. The aspect of content recommendation is reflected by the fact that the optimization decision is to select a subset of content files that tends to maximize the total user interest; the aspect of content caching is reflected by the constraints of storage space S and backhaul capacity δ; the aspect of content delivery is reflected by modelling the utility function to be dependent on the content delivery delay, which is a major performance metric for content delivery systems. Our problem formulation unifies all the key parameters that appear in the processes of content recommendation, caching, and delivery.

Two difficulties arise in solving this optimization problem. First, the expectation operation appeared within the utility function should be analytically evaluated for tractability; second, the problem appears to be a nonlinear 0–1 integer programming problem, which is non-deterministic polynomial-time hard (NP-hard) and difficult to solve directly. In what follows, we will first deal with the first difficulty by proposing an analytical approximation for the utility function.

3.2. Analytical Approximation of the Utility Function

Small cell users typically enjoy a high SINR. Considering the high SINR regime, we can approximate Equation (4) as

$$
C_k = \frac{W}{U \lg 2} \lg \gamma_k + \frac{W}{U \lg 2} \lg g_m. \tag{10}
$$

Let $q = -C_k$, we can show that q follows a generalized extreme value (GEV) distribution with three parameters given by $\mu = -W \lg \gamma_k / U \lg 2$, $\sigma = W/U \lg 2$, $\xi = 0$. The CDF of q is

$$
F_q(x) = \exp\{-\exp[-Ux \lg 2/W - \lg \gamma_k]\}. \tag{11}
$$

Given Equation (11), the expectation of $Z_{k,f}(n)$ defined in Equation (6) can be evaluated as

$$
\begin{aligned}
\mathbf{E}\{Z_{k,f}(n)\} &= \xi P(L_f/C_k(n) \leq T_0) \\
&= \xi P(L_f/T_0 \leq C_k(n)) \\
&= \xi P(-C_k(n) \leq -L_f/T_0) \\
&= \xi F_q(-L_f/T_0) \\
&= \xi \exp\left\{-\exp\left[U(n) L_f \lg 2/T_0 W - \lg\left[\gamma_k(n)\right]\right]\right\}.
\end{aligned} \tag{12}
$$

Substituting Equation (12) into Equation (8), the utility function can be evaluated in closed-form.

3.3. Problem Variations with Cross-Layer Consideration

The original optimization problem (P_1) implies that the optimizer has full cross-layer information, which not only includes higher layer (i.e., application layer) information such as the content popularity profile, content length profile, and user interest profile, but also includes lower layer information such as user association (information at the medium access control (MAC) layer) and user channel path-loss (information at the physical layer). It is worth noting that the cross-layer analytical framework proposed in this paper is different from the conventional frameworks of cross-layer analysis [35,36]. As shown in Figure 1, both ours and the conventional frameworks consider multiple protocol stacks from the top application layer to the bottom physical layer. However, conventional frameworks only address the data-level of the application layer, while our framework goes one step further to address the semantic-level of the application layer by bringing content recommendation into the design space. As a result, our problem formulation resembles a classic recommendation problem [37], and differs from the network utility maximization problem usually appeared in conventional cross-layer literature [38]. Unification of these two types of frameworks is also a promising area for future research. In reality, however, it is difficult or even impractical to obtain the exact lower layer information. Depending on the capability of cross-layer sensing and the availability of lower layer information, we can further distinguish the following three cases.

Case 1: In this case, both the user association (i.e., which users are currently served by the BS) and user SINR are known. This is the ideal case with full cross-layer information. The utility function is then given by

$$V_f^{c1}(n) = \frac{1}{U(n)} \sum_{k \in \mathbb{U}} I_{k,f} \xi \exp\left\{ -\exp\left[U(n) L_f \lg 2 / T_0 W - \lg\left[\gamma_k(n)\right] \right] \right\}. \tag{13}$$

Case 2: In this case, the user association is known, but the user SINR γ_k is unknown. However, we assume that the average SINR $\overline{\gamma}$ of all users is known. It follows that the γ_k in Equation (12) should be replaced by $\overline{\gamma}$. The utility function should be changed correspondingly into

$$V_f^{c2}(n) = \frac{1}{U(n)} \sum_{k \in \mathbb{U}} I_{k,f} \xi \exp\left\{ -\exp\left[U(n) L_f \lg 2 / T_0 W - \lg\left(\overline{\gamma}\right) \right] \right\}. \tag{14}$$

Case 3: In this case, both the user association and user SINR are unknown. However, the number of users and the average SINR is known. The utility function should now consider all users and is given by

$$V_f^{c3} = \frac{1}{K} \sum_{k=1}^{K} I_{k,f} \xi \exp\left\{ -\exp\left[K L_f \lg 2 / T_0 W - \lg\left(\overline{\gamma}\right) \right] \right\}. \tag{15}$$

We can see that Case 1 has a 'true' utility function and yields the global optimal performance. On the contrary, Cases 2 and 3 have biased utility functions, and hence their solutions may be distorted. If all the three cases are evaluated according to the same 'true' utility function, we can expect to see some performance gaps between Case 1 and Case 2 /Case 3. Such performance gaps will indicate the usefulness of different sources of cross-layer information. The optimization problems of these three cases have the same structure as the original problem of (P_1), which appears to be a difficult nonlinear combinatorial optimization problem. In the following section, we will introduce the algorithms to solve (P_1).

4. Optimal and Heuristic Content Placement Algorithms

4.1. Problem Linearization and Optimal Algorithm

The nonlinearity of optimization problem in (P_1) comes from the second constraint, which is imposed by limited backhaul capacity. Let us introduce a new variable $y_f(n){=}x_f(n-1)x_f(n)$. Based on this definition, we can write

$$\sum_{f=1}^{F} L_f x_f(n) - \sum_{f=1}^{F} L_f y_f(n) \leq \delta. \tag{16}$$

According to the very definition of $y_f(n)$, we introduce two additional linear constraints as follows:

$$\begin{aligned} &y_f(n) \leq x_f(n-1), \\ &y_f(n) \leq x_f(n). \end{aligned} \tag{17}$$

Substituting Equations (16) and (17) into (P_1), the original multi-slot content placement problem can be transformed into a linear 0–1 integer programming problem given by

$$\begin{aligned} (P_2) \quad &\max_{x} \sum_{n=1}^{N} \sum_{f=1}^{F} V_f(n) x_f(n) \\ &st. \sum_{f=1}^{F} L_f x_f(n) \leq S, \forall n \\ &\sum_{f=1}^{F} L_f x_f(n) - \sum_{f=1}^{F} L_f y_f(n) \leq \delta, \forall n \\ &y_f(n) \leq x_f(n-1) \\ &y_f(n) \leq x_f(n) \\ &x_f(n) \in \{0,1\} \\ &y_f(n) \in \{0,1\}. \end{aligned} \tag{18}$$

We note that in (P_2), x_f(n) and y_f(n) are all treated as decision variables in the optimization problem. We will subsequently prove that (P_2) is equivalent to (P_1).

Proof. The key is to demonstrate that y_f(n) = $x_f(n-1)x_f(n)$ will always hold for the optimum solutions of (P_2), so that the newly-introduced variable in (P_2) does not change the decision space of (P_1). The proof starts by considering two complementary cases.

In the first case, either x_f(n) or $x_f(n-1)$ is zero. According to Equation (17), we have $y_f(n) = 0$. In this case, $y_f(n) = x_f(n-1)x_f(n)$ holds true, and hence (P_2) is equivalent to (P_1).

In the second case, both $x_f(n-1)$ and $x_f(n)$ are equal to 1. We have $y_f = 1$ in (P_1), but $y_f(n)$ can be either 0 or 1 in (P_2) according to Equation (18). We can then write the second constraint in (P_2) as

$$\sum_{f=1}^{F} L_f x_f(n) \leq \delta + \sum_{f=1}^{F} L_f y_f(n). \tag{19}$$

However, because $y_f(n)$ is an optimization variable in (P_2), we can see that $y_f(n) = 1$ is always a better solution than $y_f(n) = 0$ in this case. Consequently, the optimization process will ensure $y_f(n) = 1$, which means $y_f(n) = x_f(n-1)x_f(n)$ holds true for the optimal solution of (P_2), and hence (P_2) is equivalent to (P_1). $\square$

Because (P_2) is linear, the optimal algorithm to solve the problem is the classic branch and bound algorithm [39–41]. However, due to the combinatorial nature of the problem (P_2), the complexity of the optimal algorithm scales exponentially with the problem size (i.e., the number of files). In the special case of $N = 1$, the original multi-slot content placement problem is simplified to a single-slot content placement problem. It is easy to show that the single-slot problem is a classic two-dimensional 0–1 knapsack problem, which can be solved by dynamic programming with a pseudo-polynomial time complexity [42]. In practice, however, the pseudo-polynomial complexity can still become exponentially complex in the worse case. In the context of edge computing, it is desirable to have low-complexity algorithms with a strict polynomial time complexity. To this end, low-complexity heuristic algorithms will be introduced subsequently.

4.2. Heuristic Algorithms

4.2.1. Heuristic Algorithm for the Single Time Slot Problem

We first present a heuristic algorithm for the single time-slot problem (i.e., $N = 1$). This algorithm will serve as the basis for the heuristic algorithm used for a multiple time slot problem. The proposed algorithm is based on a simple greedy heuristic: cache files with the largest file value per unit length until the cache space or the backhaul capacity exceed the constraints. The core of this algorithm is a sorting operation, hence the algorithm has a polynomial complexity given by $O(F \log F)$.

4.2.2. Heuristic Algorithm for the Multiple Time Slot Problem

A major drawback of the single time slot algorithm is that the "ping-pong" phenomenon may occur, which means that a file is cached in slot n, deleted in slot $n + 1$, and cached again in slot $n + 2$. This phenomenon will impose an unnecessary burden on the backhaul and reduce the overall caching performance. Compared with the single time slot optimization, a major advantage of multi-slot optimization is to eliminate the ping-pong phenomenon. This inspires us to introduce a heuristic algorithm. The rationale is to first perform independent single slot optimization for three consecutive time slots. If the ping-pong phenomenon is observed (i.e., find a file whose decision variable is 1-0-1 for three consecutive slots), eliminate the phenomenon by enforcing the decision variable to 1-1-1. Our heuristic algorithm operates on three time slots because at least three time slots are required to detect the ping-pong phenomenon. The pseudo-code of this algorithm is shown in Algorithm 1. Similar to the single-slot algorithm, Algorithm 1 also has a polynomial complexity given by $O(F \log F)$.

Algorithm 1 Heuristic algorithm for the multi-slot problem

```
Input: $x'$, $x$, $v$,$w$, $\sigma$, $n$, $I$
Output: $x_{op}$
1: Initialization: Let x′ denotes decision variables of the last slot, $x$ is the decision variables of the current slot, $v$ represents the file value per unit length, $w$ and $\sigma$ indicates the cache space and update threshold, $n$ is the number of files, $I$ indicates maximum iteration number and $i$ indicates the iteration index
2: Step1: Obtain decision variables of three consecutive slots with the optimal algorithm in single-slot senario and the overall cache value $V$
3: Step2: Find a file set $\Phi$ whose decision variable is 1-0-1 for three consecutive slots
4: if $\Phi = \emptyset$ then
5:     Go to Step6
6: Step3:
7: if $I < i$ then
8:     Go to Step6
9: Step4: Randomly choose a file $f \in \Phi$ and change its decision variable to 1-1-1, then use heuristic algorithm for the single time slot problem to recalculate the decision variables for the remaining files in the last two slots and overall value $V'$
10: Step5:
11: if $V \leq V'$ then
12:     Update $x_{op} = x$ and $V = V'$
13: else
14:     Change decision variables back to their original ones and return to Step3.
15: Step6: return optimal decision variables $x_{op}$
```

5. Theoretical Performance Bounds with Simplified Model

The value of an explicit caching system can be evaluated by the expected user QoE. So far, although algorithms to maximize the user QoE have been obtained, we still lack a clear analytical insight into how the user QoE is related to various system parameters. This section aims to derive some theoretical performance bounds of the explicit caching system. For analytical tractability, we will apply the simple content model introduced in Section 2.2. Moreover, the following simplifying assumptions are further made in this section.

- The system performance is evaluated according to the utility function presented by Equation (14), which corresponds to Case 2. We recall that this case assumes that the user association and average user SINR is known. In this section, we will treat the number of users and the average SINR as fixed value parameters and denote them as U and $\bar{\gamma}$, respectively. Although this section focuses on Case 2 only, we will show the latter in Section 6 that the performance of Case 2 serves as a very good predictor (almost identical) to the ideal performance in Case 1.
- The cache space constraint S is replaced by a file number constraint M, which limits the maximum number of files that can be cached. This simplification is reasonable because the value of the cached content tends to concentrate on a subset of high-value files. In other words, a fixed number of high-value files will capture most of the value of the entire cached file set. This simplification implies that our analysis in this section should be interpreted as an approximation.

The above assumptions yield a new optimization problem given by

$$\begin{aligned}
(P_3) \quad & Q = \max_x \sum_{n=1}^{N} \sum_{f=1}^{F} V_f^{c2}(n) x_f(n) \\
& st. \sum_{f=1}^{F} x_f(n) = M, \forall n \\
& \sum_{f=1}^{F} L_f x_f(n) - \sum_{f=1}^{F} L_f y_f(n) \leq \delta, \forall n \\
& y_f(n) \leq x_f(n-1) \\
& y_f(n) \leq x_f(n) \\
& x_f(n) \in \{0,1\} \\
& y_f(n) \in \{0,1\}.
\end{aligned} \tag{20}$$

This new problem, which is analytically tractable, can serve as an approximation to the original problem.

5.1. CDF of the Utility Function

The first step is to evaluate the CDF of the utility function $V_f^{c2}(n)$, which can be written as the product of two independent random variables

$$V_f^{c2}(n) = XY, \tag{21}$$

where

$$\begin{aligned}
X &= c\exp[-\exp(aL_f + b)], \\
Y &= \sum_{k \in \mathbb{U}} I_{k,f}.
\end{aligned} \tag{22}$$

Here, we have $c = \xi / U(n)$, $a = U(n) \lg 2 / T_0 W$, and $b = -\lg \overline{\gamma}$.

Let us first focus on the CDF of the random variable X. Given uniform file size distribution, the CDF of the random file length L_f is given by

$$F_{L_f}(l) = \begin{cases} 0, & l < l_{min}, \\ \dfrac{l - l_{min}}{l_{max} - l_{min}}, & l_{min} \leq l \leq l_{max}, \\ 1, & l > l_{max}. \end{cases} \tag{23}$$

Because X is a function of L_f, it follows that CDF of X can be derived as

$$\begin{aligned}
F_X(x) &= P(X \leq x) \\
&= \mathbb{P}\left(\exp[-\exp(aL_f + b)] \leq \frac{x}{c}\right) \\
&= \mathbb{P}\left(\exp[aL_f + b] \geq \ln(\frac{c}{x})\right) \\
&= \mathbb{P}\left(L_f \geq \frac{1}{a}\ln\left(\ln(\frac{c}{x})\right) - \frac{b}{a}\right) \\
&= 1 - \mathbb{P}\left(L_f \leq \frac{1}{a}\ln\left(\ln(\frac{c}{x})\right) - \frac{b}{a}\right) \\
&= 1 - F_{L_f}\left(\frac{1}{a}\ln\left(\ln(\frac{c}{x})\right) - \frac{b}{a}\right) \\
&= \begin{cases} 0, & x < x_{min} \\ 1 - \dfrac{\ln\left(\ln(\frac{c}{x})\right) + al_{min} + b}{a(l_{max} - l_{min})}, & x_{min} \leq x \leq x_{max} \\ 1, & x > x_{max}, \end{cases}
\end{aligned} \tag{24}$$

where $x_{min} = c\exp[-\exp(al_{max} + b)]$, $x_{max} = c\exp[-\exp(al_{min} + b)]$.

Let us now consider the random variable Y. For a total of U users linked to the BS, the probability for an arbitrary user to be interested in the f-th file (i.e., $I_{k,f} = 1$) is

$$P = \frac{A_f}{U}, \tag{25}$$

where A_f is the aggregated user interest of the f-th file. Because the user interest profiles are independent, random variable Y obeys a binomial distribution given by $Y \sim Bi(U, P)$.

The utility function $Z = V_2^{c2}$ is a product of random variables X and Y, hence its CDF can be derived as

$$F_Z(z) = \sum_{y_i>0} p_i F_X\left(\frac{z}{y_i}\right) + \sum_{y_i<0} p_i\left(1 - F_X\left(\frac{z}{y_i}\right)\right) + p_{y_i=0} F_X\left(\frac{z}{0}\right). \tag{26}$$

Because random variable Y is nonnegative, we can omit the situation when $y_i < 0$. It follows that Equation (26) can be further refined as

$$\begin{aligned} F_Z(z) &= \sum_{y_i>0} p_i F_X\left(\frac{z}{y_i}\right) + p_{y_i=0} F_X\left(\frac{z}{0}\right) \\ &= \sum_{i=1}^{U} \binom{U}{i} P^i (1-P)^{U-i} F_X\left(\frac{z}{i}\right) + \binom{U}{0} P^0 (1-P)^U. \end{aligned} \tag{27}$$

5.2. Bounds of Average User QoE Performance

Once the CDF of the utility function is known, we can move on to analyze the outcome Q (i.e., optimized average user QoE) of the optimization problem defined $(P3)$. We distinguish two extreme cases that correspond to the lower bound and upper bound of the optimized QoE performance.

In case of the lower bound, the optimization problem $(P3)$ is solved by a random caching policy, which randomly choose M files from the content pool. Such a random policy is optimal in two situations: during initialization or when the backhaul capacity is zero. This is because both situations should assume that all users are equally probable to be served by the BS over the long term, and hence each piece of content would have the same value over the long term. It follows that the expected outcome can be evaluated as

$$V_{low} = M\widetilde{V}, \tag{28}$$

where $\widetilde{V}$ represents the expected value of a randomly chosen file, which can be calculated as

$$\widetilde{V} = \int z f_Z(z) dz, \tag{29}$$

where $f_Z(z)$ (i.e., PDF) is the first-order differentiation of $F_Z(z)$.

In case of the upper bound, the optimization problem is solved by a greedy policy, which always picks the most valuable M files. The policy is feasible when the backhaul capacity is unlimited (or large enough to support any update). According to the results of order statistics [43–45], given F i.i.d., random variables with CDF denoted by $F(x)$, the CDF of the r-th largest random variable is given by

$$G_{(r)}(x) = \sum_{i=r}^{F} \binom{F}{i} F^i(x)[1 - F(x)]^{F-i}. \tag{30}$$

The PDF of the r-th order statistics is

$$g_{(r)}(x) = \frac{dG_{(r)}(x)}{dx}. \tag{31}$$

The expectation of the r-th order random variable is

$$\mathbf{E}\{X_r\} = \int_{x_{min}}^{x_{max}} x g_{(r)}(x) dx, \tag{32}$$

where $g_{(r)}(x)$ is the corresponding probability density function (PDF) of $G_{(r)}(x)$. The expectation of the most valuable M files can be calculated as

$$\begin{aligned} V_{up} &= \mathbf{E}\{X_{F-M+1} + X_{F-M+2} + ... + X_F\} \\ &= \mathbf{E}\{X_{F-M+1}\} + \mathbf{E}\{X_{F-M+2}\} + ... + \mathbf{E}\{X_F\} \\ &= \int_{x_{min}}^{x_{max}} x[g_{(F-M+1)}(x) + g_{(F-M+2)}(x) + ... + g_{(F)}(x)]dx. \end{aligned} \tag{33}$$

5.3. Validation of the Theoretical Bounds

In this subsection, we carry out Monte Carlo simulations to validate the derived theoretical performance bounds of the explicit caching system. In our simulation, the total number of users is set to be 100, while the average number of users served by the BS is set to be 10. The available bandwidth W = 10 MHz and the maximum user delay tolerant is 1 s. The total number of files F is 1000. The files size follows uniform distribution in [1, 9]. Numerical results are obtained based on direct calculations from the derived equations, while empirical results are obtained by solving 100 independent (P3) optimization problems and taking numerical average over the 100 snapshots of optimal outcomes.

Figure 3 shows the upper and lower bounds of the expected QoE as functions of the number of cached files M. It can be observed that the lower bound increases almost linearly with increasing M, while the upper bound increases with diminishing returns when M gets large. Moreover, the numerical (i.e., theoretical) and empirical (i.e., simulation) curves are shown to agree well. This serves as a validation to our theoretical derivations.

Figure 4 shows empirical results of the expected QoE as functions of the normalized backhaul capacity. Numerical/theoretical results on the upper and lower bounds of the QoE are also presented. It can be observed that, when the backhaul capacity gradually increases from zero to a large value, the empirical performance changes from the theoretical lower bound to the theoretical upper bound. This confirms our previous statement that the derived lower and upper bounds correspond to the two extreme cases of zero backhaul capacity and unlimited backhaul capacity, respectively.

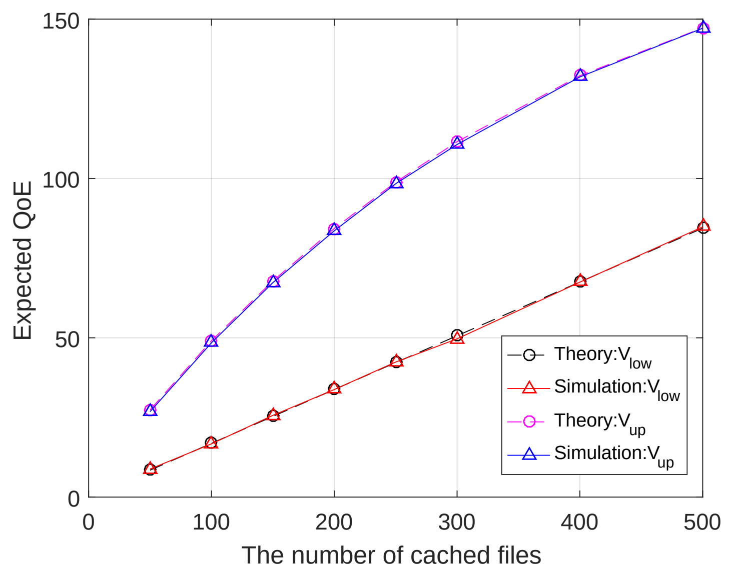

Figure 3. The expected QoE as a function of the number of cached files M.

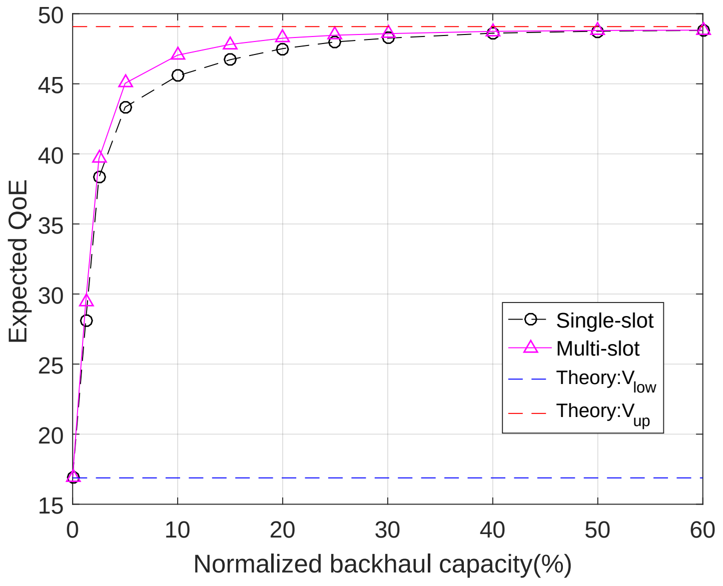

Figure 4. The expected QoE as a function of the normalized backhaul capacity (M = 100).

6. Simulation Results and Discussions with Realistic Model

In this section, Monte Carlo simulations are performed to evaluate the performance of the explicit caching system with realistic modelling assumptions. The average number of users served by the BS is set to be 10 and the available bandwidth W is 10 MHz. The total number of users is set to be 100 and the maximum user delay tolerant is 1 s.

We consider a mixed content library containing two heterogeneous types of files: texts and videos. Different types of contents have different characteristics in terms of the file length distribution and user interest profile. Our simulations consider a library consisting of 500 text files and 500 video files. The lengths of text files and video files follow two log-normal distributions with mean values of 1 MB and 10 MB, respectively. The popularity of files follows a Zipf distribution with $\beta = 1$ and $Z_{max} = 80$. The users' interest for a specific file follows a half-normal distribution with unit variance. We further assume that the QoE of video is 10 times that of texts, i.e., $\xi = 1$ for texts and $\xi = 10$ for videos. The SINR of users is assumed to be exponentially distributed with a mean value of 10 dB and truncated to have a minimum SINR of 3 dB.

Given the above settings, our simulation runs 100 snapshots. In each snapshot, the user interest profile, user association, and user channel state information, etc. are generated independently and randomly. The simulation results are averaged over the 100 snapshots.

6.1. Policies with Cross-Layer Information

This subsection aims to answer the following question: to what extent can cross-layer information contribute to improving the performance of explicit caching. Without loss of generality, our discussion is constrained to the single-slot optimization problem (i.e., $N = 1$). We consider six different caching policies categorized into two types. The first type is policies that exploit physical or MAC layer information. We have three policies of this type, corresponding to the three cases discussed in Section 3. The second type is conventional caching policies that rely solely on upper layer information. Three representative policies of the second type are considered: the most popular content (MPC) policy [46], largest content diversity (LCD) policy [47], and largest popularity per unit content (PPU) policy, which caches contents based on the criteria of having large popularity per unit length. We note that the first type policies use optimal algorithms with exponential complexity, while the second type policies are greedy algorithms with polynomial complexity.

Figure 5 shows the expected user QoE as a function of the storage space when different caching policies are applied. The storage space is normalized by the total size of the file library. The backhaul capacity limit is set to be 50% of the storage space. Figure 5 shows that increasing the storage space can improve the QoE performance but has diminishing returns. Moreover, policies of the first type significantly outperform all policies of the second type. Among the three policies of the first type, Case 1 and Case 2 policies yield almost identical performance, while Case 3 gives a slightly worse performance. This suggests that the advantage of the first type policy mainly comes from exploiting the information of user association rather than the information of user SINR.

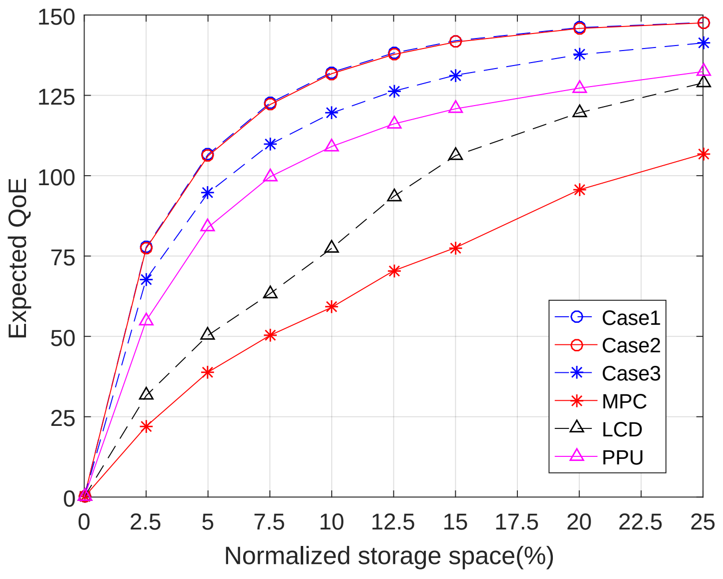

Figure 5. The expected QoE as a function of normalized storage space for different caching policies.

6.2. Impacts of Backhaul, Mobility, and Mobility Prediction

Apart from cross-layer information, we are interested in the impacts of backhaul capacity, user mobility, and the number of jointly optimized time slots on the system performance. Without loss of generality, we choose the (near-optimal) Case 2 policy as our caching policy. Figure 6 shows the expected QoE per user as a function of the backhaul capacity. The backhaul capacity is normalized by the maximum backhaul capacity, which is set to be 500 MB. We consider two cases with $N = 1$ (single-slot) and $N = 5$ (multi-slot). Moreover, we consider two scenarios with high user mobility ($P_a = 0.8$) and low user mobility ($P_a = 0.2$).

Figure 6 reveals that high user mobility generally demands greater backhaul capacity, which is an intuitive result. However, it also shows that, when the backhaul capacity is very small or very large, the performances with high mobility and low mobility converge. This implies that mobility and backhaul capacity are relevant, but not limiting factors that will constrain the fundamental system performance. The system performance is fundamentally constrained by other factors such as bandwidth, content interest profile, and cache storage size, which can be easily seen from the utility function and storage constraint. Moreover, Figure 6 shows that the multi-slot algorithm outperforms the single-slot algorithm. We note that applying the multi-slot algorithm implies that users' mobility (and hence user association) in $N = 5$ time slots can be accurately predicted. The results of Figure 6 suggest that mobility prediction can be traded for backhaul capacity, but has no impact on the fundamental performance limit.

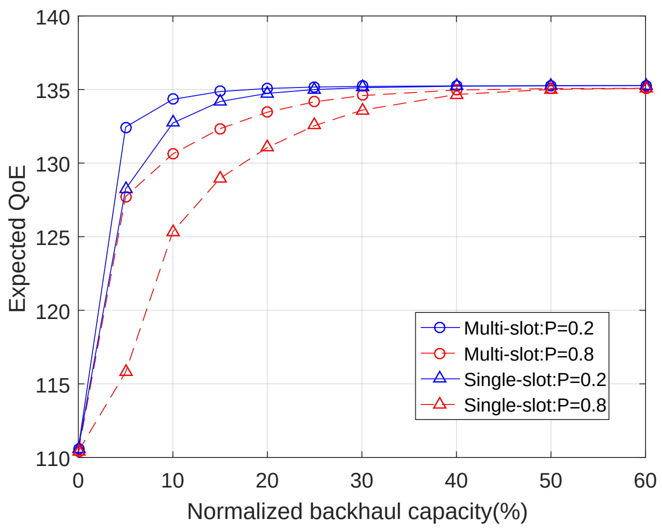

Figure 6. The expected QoE as a function of the normalized backhaul capacity, with varying user mobility and number of time slots.

6.3. Impacts of User Density and File Popularity Distribution

Figure 7 shows the expected QoE as a function of the normalized storage space with varying system parameters. First, Figure 7 illustrates the impact of the average number of users per BS on the QoE performance. It can be seen that, as the number of users U increases, the QoE performance degrades. This is because, when U increases, the downlink capacity allocated to each user reduces, which ultimately results in a reduction in the expected QoE. In addition, Figure 7 also shows how the QoE performance changes with different $R_{max}(R_{max} = Z_{max}/K)$, which represents the concentration of file popularity. It can be seen that, with increasing R_{max}, the performance of the system also increases due to a higher concentration of user interest in a given number of cached files.

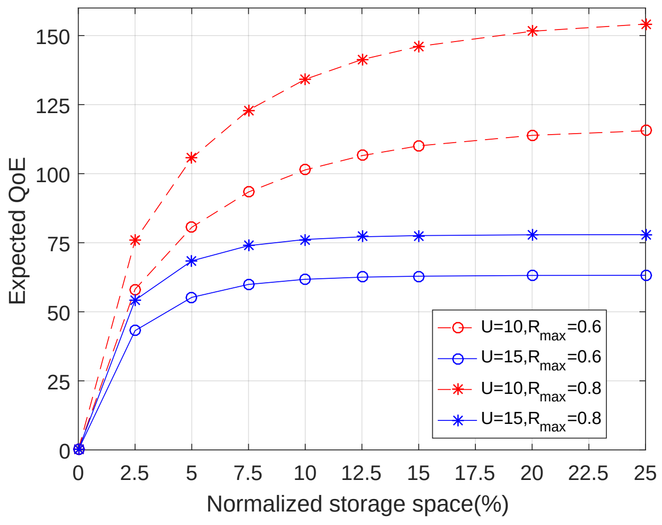

Figure 7. The expected QoE as a function of the normalized storage space, with varying values of the average number of users per BS and different distributions of the file popularity.

6.4. Impacts of High SINR Approximation

In Section 3.2, the high SINR approximation has been introduced to to compute the utility function (i.e., expected capacity) in closed-form. Alternatively, the exact value of the utility function can be obtained by the Monte Carlo method, which is computationally inefficient. To reveal the possible drawbacks of using the high SINR approximation, Figure 8 compares the system performance of two algorithms: the proposed algorithm that uses the high SINR approximation in its utility function, and the optimal algorithm that uses the Monte Carlo method (i.e., exact method) in its utility function. Results show that the high SINR approximation brings very small performance penalty even when the actual average SINR is small (e.g., 3 dB). These results suggest that the proposed algorithm with the high SINR approximation is robust for practical ranges of SINR.

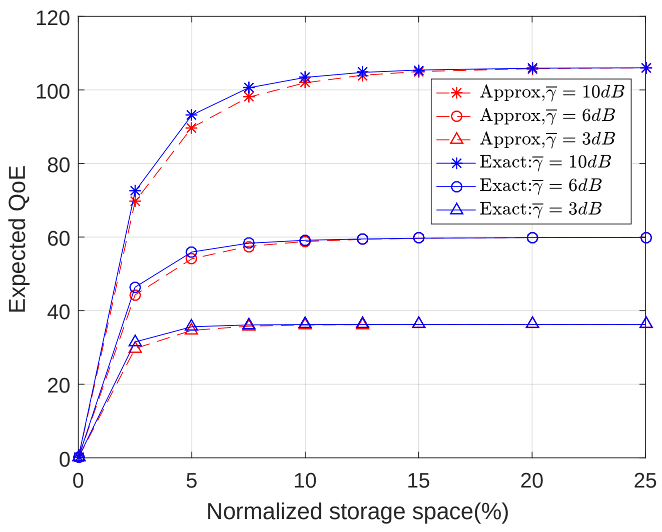

Figure 8. The expected QoE as a function of the normalized storage space with approximated and exact utility functions.

6.5. Comparison between Algorithms

In this subsection, we want to evaluate the performance gaps between the optimal algorithms and the heuristic algorithms proposed in Section 4. Figures 9 and 10 compare the performance in single-slot and multi-slot scenarios, respectively. It can be seen in Figure 9 that, in the single-slot scenario, the proposed heuristic algorithm yield almost identical performance to the optimal algorithm. In the multi-slot scenario, Figure 10 shows that the performance gaps between heuristic and optimal algorithms are small in general, and diminishes in the extreme cases when the backhaul capacity approaches zero or becomes very large. These results suggest that the proposed heuristic algorithms are effective algorithms to be used in practice.

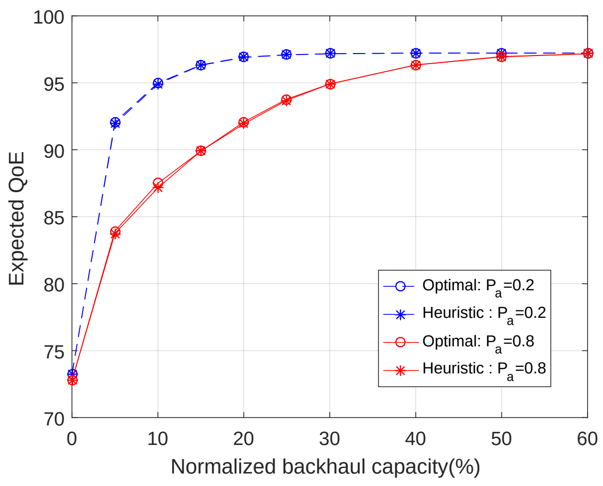

Figure 9. The expected QoE as a function of the normalized backhaul capacity in the single time slot optimization problem (S = 200 MB).

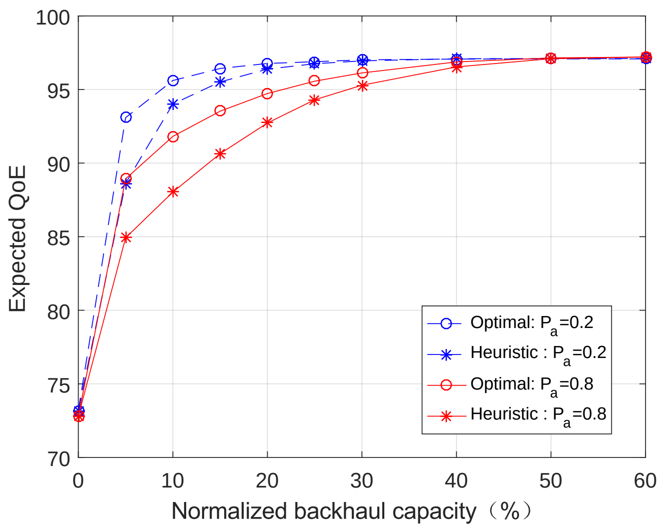

Figure 10. The expected QoE as a function of the normalized backhaul capacity in the multiple time slot optimization problem (S = 200 MB, I = 500).

6.6. Discussions on Future Work

Finally, our work in this paper considers a simplified scenario of independent BS and independent backhaul. In practice, nearby BSs may cooperate to achieve better performance, and the backhaul links may be shared among multiple BSs. The cross-layer analytical framework established in this paper can be extended. Moreover, the results obtained in this paper can be used as heuristic or initial inputs for the more complicated cases of interdependent BSs. For example, in the shared backhaul scenario, we can formulate the problem of backhaul capacity allocation as a multi-level water-filling problem, using the QoE-backhaul relationship of each BS (as showed in Figures 9 and 10 in this paper) as inputs to calculate the water levels. These promising extensions will be considered in our future work.

7. Conclusions

This paper proposes a novel BS caching paradigm called explicit caching. Optimal and near-optimal heuristic algorithms have been proposed to solve the content caching problem. Using user QoE as the ultimate performance metric, we have systematically investigated the theoretical and simulated performances of the explicit caching system. It has been revealed that the performance of an explicit caching system is fundamentally limited by the cache storage space, user interest profile, and available radio bandwidth, while increasing backhaul capacity, exploiting cross-layer information, and having user mobility prediction can only contribute to better approaching the fundamental performance bounds. We conclude that the explicit caching system is a novel and promising edge computing application for personalized content dissemination over mobile edge networks.

Acknowledgments: The authors would like to acknowledge the support from the National Natural Science Foundation of China (NSFC) (Grant No. 61571378).

Author Contributions: Lingyu Chen contributed to the original ideas and designed the simulations; Youxing Su derived the formulas, carried out the simulations; Wenbin Luo researched the literatures and revised the formulas; Xuemin Hong contributed to the scheme design and drafted the manuscriptl; Jianghong Shi revised the manuscript and analyzed the simulation results.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cisco Visual Networking Index: Global Mobile Data Taffic Forecast Update, 2016–2021. Available online: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html (accessed on 23 January 2018).

2. Shanmugam, K.; Golrezaei, N.; Dimakis, A.G.; Molisch, A.F.; Caire, G. Femto caching: Wireless content delivery through distributed caching helpers. *IEEE Trans. Inf. Theory* **2013**, *59*, 8402–8413.
3. Zhang, Y.; Bao, L.; Welling, M.; Yang, S.H. Base station localization in search of empty spectrum spaces in cognitive radio networks. In Proceedings of the International Conference on Mobile Ad-Hoc and Sensor Networks, Fujian, China, 14–16 December 2009; pp. 94–101.
4. Han, T.; Ansari, N. Network utility aware traffic load balancing in backhaul-constrained cache-enabled small cell networks with hybrid power supplies. *IEEE Trans. Mobile Comput.* **2017**, *16*, 2819–2832.
5. Biral, A.; Centenaro, M.; Zanella, A.; Vangelista, L.; Zorzi, M. The challenges of M2M massive access in wireless cellular networks. *IEEE Digit. Commun. Netw.* **2015**, *1*, 1–19.
6. Ota, K.; Dong, M.; Gui, J.; Liu, A. QUOIN: Incentive mechanisms for crowd sensing networks. *IEEE Net. Mag.* **2018**, *99*, 1–6.
7. Jararweh, Y.; Doulat, A.; AlQudah, O.; Ahmed, E.; Al-Ayyoub, M.; Benkhelifa, E. The future of mobile cloud computing: Integrating cloudlets and mobile edge computing. In Proceedings of the 2016 23rd International Conference on Telecommunications (ICT), Thessaloniki, Greece, 16–18 May 2016; pp. 1–5.
8. Hu, Y.C.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. *Mobile Edge Computing—A Key Technology towards 5G*, 1st ed.; ETSI White Ref., no. 11; European Telecommunications Standards Institute (ETSI): Sophia Antipolis, France, 2015.
9. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. a survey on mobile edge computing: The communication perspective. *IEEE Commun. Surveys Tutor.* **2017**, *19*, 2322–2358.
10. Ma, F.; Liu, X.; Liu, A.; Zhao, M.; Huang, C.; Wang, T. a time and location correlation incentive scheme for deeply data gathering in crowdsourcing networks. *Wirel. Commun. Mob. Comput.* **2018**, *8*, 1–22.
11. Ioannou, A.; Weber, S. a survey of caching policies and forwarding mechanisms in information-centric networking. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2847–2886.
12. Su Z.; Xu, Q. Content distribution over content centric mobile social networks in 5G. *IEEE Commun. Mag.* **2015**, *53*, 66–72.
13. Dong, L. Caching at the wireless edge: Design aspects, challenges, and future directions. *IEEE Commun. Mag.* **2016**, *54*, 22–28.
14. Cha, M.; Kwak, H.; Rodriguez, P.; Ahn, Y.Y.; Moon, S. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In Proceedings of the ACM SIGCOMM Conference on Internet Measurement, San Diego, CA, USA, 24–26 October 2007; pp. 1–14.
15. Gregori, M.; Gómez-Vilardebó, J.; Matamoros, J. Wireless content caching for small cell and D2D networks. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 1222–1234.
16. Cheng, J.; Jamin, S.; Raz, D. The cache location problem. In *Building Scalable Network Services*; Springer US: New York, NY, USA, 2004; pp. 67–98.
17. Poularakis, K.; Iosifidis, G.; Tassiulas, L. Approximation algorithms for mobile data caching in small cell networks. *IEEE Trans. Commmun.* **2014**, *62*, 3665–3677.
18. Chen, Z.; Kountouris, M. D2D caching vs. small cell caching: Where to cache content in a wireless network. In Proceedings of the International Conference on Wireless Communications and Signal Processing, Edinburgh, UK, 3–6 July 2016; pp. 1–6.
19. Krishnan, P.; Raz, D.; Shavitt, Y. The cache location problem. *IEEE/ACM Trans. Netw.* **2000**, *8*, 568–582.
20. Golrezaei, N.; Shanmugam, K.; Dimakis, A.G. Femto caching: Wireless video content delivery through distributed caching helpers. In Proceedings of the International Conference on Computer Communications, Turin, Italy, 14–19 April 2013; pp. 1107–1115.
21. Wang, R.; Peng, X.; Zhang, J. Mobility-aware caching for content-centric wireless networks: Modeling and methodology. *IEEE Commun. Mag.* **2016**, *54*, 77–83.
22. Pantisano, F.; Bennis, M.; Saad, W. Cache-aware user association in backhaul-constrained small cell networks. In Proceedings of the International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, Hammamet, Tunisia, 12–14 May 2014; pp. 37–42.
23. Siris, V.A.; Vasilakos, X.; Polyzos, G.C. Efficient proactive caching for supporting seamless mobility. In Proceedings of the International Symposium on a World of Wireless, Mobile and Multimedia Networks, Sydney, Australia, 19 June 2014; pp. 1–6.
24. Li, H.; Hu, D. Mobility prediction based seamless RAN-cache handover in HetNet. In Proceedings of the Wireless Communications and Networking Conference, Doha, Qatar, 3–6 April 2016.

25. Park, J.; Kim, S.L. Content-specific broadcast cellular networks based on user demand prediction: a revenue perspective. In Proceedings of the Wireless Communications and Networking Conference (WCNC), Istanbul, Turkey, 6–9 April 2014; pp. 2271–2276.
26. Pingyod, A.; Somchit, Y. Content updating method in femtocaching. In Proceedings of the International Joint Conference on Computer Science and Software Engineering, Ban Phatthaya, Thailand, 14–16 May 2014; pp. 1–6.
27. Zonoozi, M.M.; Dassanayake, P. User mobility modeling and characterization of mobility patterns. *IEEE J. Sel. Areas Commun.* **1997**, *15*, 1239–1252.
28. Bastug, E.; Bennis, M.; Debbah, M. Living on the edge: The role of proactive caching in 5G wireless networks. *IEEE Commun. Mag.* **2014**, *52*, 82–89.
29. Chandrasekhar, V.; Andrews, J.G.; Gatherer, A. Femtocell networks: a survey. *IEEE Commun. Mag.* **2008**, *46*, 59–67.
30. Zhang, Y.; Lu, H.; Wang, H.; Hong, X. Cognitive cellular content delivery networks: Cross-layer design and analysis. In Proceedings of the Vehicular Technology Conference (VTC Spring), Nanjing, China, 15–18 May 2016; pp. 1–6.
31. Falah, A.; Luo, W.; Jiao, J.; Tang, B.; Hong, X.; Shi, J. Explicit data caching at mobile edge networks: A cross-layer perspective. In Proceedings of the IEEE SPAWC 2017, Saporo, Japan, 2–7 July 2017; pp. 1–6.
32. Cai, T.; Cai, H.J.; Zhang, Y.; Huang, K.; Xu, Z. Polarized score distributions in music ratings and the emergence of popular artists. In Proceedings of the Science and Information Conference, London, UK, 7–9 October 2013; pp. 472–476.
33. Barford, P.; Crovella, M. Generating representative web workloads for network and server performance evaluation. In *ACM SIGMETRICS Performance Evaluation Review*; ACM: New York, NY, USA, 1998; Volume 26, pp. 151–160.
34. Andrews, J.G.; Baccelli, F.; Ganti, R.K. a tractable approach to coverage and rate in cellular networks. *IEEE Trans. Commun.* **2011**, *59*, 3122–3134.
35. Srivastava, V.; Motani, M. Cross-layer design: a survey and the road ahead. *IEEE Commun. Mag.* **2005**, *43*, 112–119.
36. Kawadia, V.; Kumar, P.R. a cautionary perspective on cross-layer design. *IEEE Wirel. Commun.* **2005**, *11*, 3–11.
37. Balabanović, M.; Shoham, Y. Fab: Content-based, collaborative recommendation. *Commun. ACM* **1997**, *40*, 66–72.
38. Pham, Q.V.; Hwang, W.J. Network utility maximization based congestion control over wireless networks: A survey and potential directives. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1173–1200.
39. Lawler, E.L.; Wood, D.E. Branch-and-bound methods: A survey. *Oper. Res.* **1966**, *14*, 699–719.
40. Mitten, L.G. Branch-and-bound methods: General formulation and properties. *Oper. Res.* **1970**, *18*, 24–34.
41. Gupta, O.K.; Ravindran, A. Branch and bound experiments in convex nonlinear integer programming. *Manag. Sci.* **1985**, *31*, 1533–1546.
42. Pisinger, D. Algorithms for knapsack problems. *N-H. Math. Stud.* **1995**, *132*, 213–257.
43. Yang, S.S. General distribution theory of the concomitants of order statistics. *Ann. Stat.* **1977**, *5*, 996–1002.
44. Kamps, U. a concept of generalized order statistics. *J. Stat. Plan. Inference* **1995**, *48*, 1–23.
45. Moghadam, S.A.; Pazira, H. The relations among the order statistics of Uniform distribution. *Trends Appl. Sci. Res.* **2011**, *6*, 719–723.
46. Zhang, N.; Guan, J.; Xu, C.; Zhang, H. a dynamic social content caching under user mobility pattern. In Proceedings of the Wireless Communications and Mobile Computing Conference (IWCMC), Nicosia, Cyprus, 4–8 August 2014; pp. 1136–1141.
47. Peng, X.; Shen, J.C.; Zhang, J.; Letaief, K.B. Backhaul-aware caching placement for wireless networks. In Proceedings of the Global Communications Conference (GLOBECOM), San Diego, CA, USA, 6–10 December 2015; pp. 1–6.