A Profit-Oriented Cooperative Caching Algorithm for Hierarchical Content Centric Networking

Meiyi Yang[1], Mingchuan Zhang[1], Junlong Zhu[1], Ruoshui Liu[1], Qingtao Wu[1], Ian J. Wassell[2]

[1] College of Information Engineering, Henan University of Science and Technology, Luoyang, 471023, China
[2] Computer Laboratory, University of Cambridge, CB3 0FD, UK
** E-mail: 160304300189@stu.haust.edu.cn; zhang_mch@haust.edu.cn; jlzhu@bupt.edu.cn; liuruoshui@msn.com; wqt8921@haust.edu.cn, ijw24@cam.ac.uk*
Corresponding author: Mingchuan Zhang

Abstract: Cooperative caching among nodes is a hot topic in Content Centric Networking (CCN). However, the cooperative caching mechanisms are performed in an arbitrary graph topology, leading to the complex cooperative operation. For this reason, hierarchical CCN has received widespread attention, which provides simple cooperative operation due to the explicit affiliation between nodes. In this paper, we propose a heuristic cooperative caching algorithm for maximizing the average provider earned profit under the two-level CCN topology. This algorithm divides the cache space of control nodes into two fractions for caching contents which are downloaded from different sources. One fraction caches *duplicated* contents and the other caches *unique* contents. The optimal value of the split factor can be obtained by maximizing the earned profit. Furthermore, we also propose a replacement policy to support the proposed caching algorithm. Finally, simulation results show that the proposed caching algorithm can perform better than some traditional caching strategies.

1 Introduction

Content Centric Networking (CCN) is an emerging network model with some innovations, such as location-independent naming, node-based caching, name-oriented routing and received-driven communication. A key component of CCN is ubiquitous caching nodes, which cache some popular contents for future demands. It is very useful to earn higher profit for content provider (thereby leading to lower contents provisioning cost) by caching the contents close to the user. Since the finite caching space of the edge nodes, it is necessary to design a good caching policy, which can make full use of all cache nodes, and then it has a positive effect on earned profit [1]–[5].

So far, the caching policies are mainly divided into two types: non-cooperative caching policies and cooperative caching policies [6]–[9]. For non-cooperative caching policies, every node caches contents independently. In this case, every node does not know the cache status of other nodes, leading to a high content redundancy and a low hit ratio in the whole network. For cooperative caching policies, every node caches content and forwards requests according to the cache status of other nodes. In this case, the requests will be faster answered, and the provider earned profit will be improved, e.g., bandwidth, delay, and etc. In spite of these advantages which are beneficial to provider, it also brings some disadvantages if these cooperative caching policies are employed in an arbitrary graph topology. For example, making caching and forwarding decisions via exchanging messages among nodes is very slow. The reason is that every node without the explicit affiliation in this topology incurs complex cooperative operations. Against that, the cooperative operation in a hierarchical topology can be simplified. Since every node has explicit affiliation, making decisions of caching contents and forwarding requests is very rapid [10]. To this end, cooperative caching policy in a hierarchical topology is considered.

Cooperative caching policies in a hierarchical topology have attracted widespread attention from the networking community [10]–[14]. To enhance the caching and forwarding contents efficiency, the works in [10] and [11] propose collaborative caching with a request routing policy. When a node forwards a request, it needs to refer to the caching information table recorded in the node and forward the request to the corresponding node. This policy improves the response speed when users request the contents. In [12], cooperative caching takes place in the neighboring mobile devices of every Social Wireless Networks (SWNET) partition. In [13], hierarchical caching policy is used in cellular backhaul network. In [14], the authors propose a combined caching-transmission policy, which is a tradeoff between transmission diversity and content diversity. The above policies have a common feature that they can not cooperate between small cells.

In this paper, our algorithm is implemented on a two-level CCN topology, which consists of control nodes and common nodes. One control node and multiple common nodes form a small base system (SBS). In this topology, content caching is performed on every SBS under the management of the control node to achieve our optimal objective — the maximization of the average earned profit. We formulate the problem with this optimal objective, which is influenced by a two-fold benchmark case of maximizing the local hit ratio in the SBSs and maximizing the hit ratio in the whole network. It has been proved that this is a NP-hard problem [15]–[17], and thus we propose a heuristic collaborative caching algorithm to attain the optimal content placement with the objective of maximizing the network-wide provider earned profit. The optimal content placement lives somewhere between the above two-fold benchmark case. In [18], the cooperative caching policy is used in SD-RAN [19], [20], which is referred to in our algorithm. In our algorithm, we divide the cache space of control nodes into two fractions. The first fraction caches the most popular contents to improve the local hit ratio in the SBSs. The second fraction caches different content coming from the server, which improves the hit ratio in the whole network. Furthermore, we can obtain the maximum average earned profit through finding the optimal value of the split factor.

The main contributions of this paper can be summarized as follows:

- To avoid the complex cooperative operation in an arbitrary graph, a two-level CCN topology is employed. According to the twofold benchmark case, we formulate the optimal objective of maximizing the average earned profit as an optimal content placement problem.

• We theoretically analyze the optimal content placement problem under a homogenous request model. Inspired by this, we propose a heuristic caching algorithm which divides the cache space of control node into two fractions, and propose a replacement policy to support this caching algorithm.
• Finally, we implement a simulation for assessing the performance of our algorithm. The simulation results verify that our algorithm outperforms some traditional strategies on maximizing the earned profit.

The rest of this paper is organized as follows. In Section 2, we introduce the related work and motivation. System model that includes network model and profit model is described in Section 3. The formulation of the twofold benchmark problem and the optimal objective are presented in Section 4, while in Section 5 we propose a heuristic caching algorithm to attain the optimal content placement with the objective of maximizing the network-wide provider earned profit, and propose a replacement policy of control node. Section 6 assesses the algorithm performance by means of simulation. Finally, the conclusions are drawn in Section 7.

2 Related Work And Motivation

Content caching is an useful method to improve the user experience and content transmission performance of the Internet. In recent years, a large body of work has been devoted to content caching in traditional overlay network architectures, such as web-caching [21], CDNs [22] and (P2P) [23]. However, these traditional overlay network caching systems exhibit fundamental differences from the in-network caching of CCN. Firstly, the caching capacity of nodes in CCN is usually finite while the caching capacity of nodes in a general caching network is always very large. Secondly, the caching nodes in CCN are widely deployed in an arbitrary graph topology, while the caching nodes in a general caching network are widely deployed in a hierarchy topology. More importantly, the existing caching policies are with respect to a special application in traditional overlay networks architectures, while in-network caching is an inherent capability and irrelevant to application in CCN. Therefore, the previous caching policies widely employed in overlay network architectures can not be directly applied in CCN.

There are many problems concerning in-network caching policies in CCN. In most CCN caching policies, a given content is cached along an en-route way (i.e., nodes on the paths from a requesting node to the one or multiple serving nodes). In this case, the feature of universal caching in CCN cannot be efficient leveraged in terms of improving content transmission performance due to the content redundancy and filter effects of multihop caches [24], [25]. In order to address this problem, some cooperative caching policies are proposed, such as [10], [11] [28] and [29]. These policies not only consider en-route notes but also nodes that are near en-route nodes to cache the requested contents.

Cooperative caching policies can be further divided into two types: centralized cooperative caching policies and distributed cooperative caching policies. Centralized cooperative caching policies select a node as the logical control plane which monitors the network status and makes caching decisions. Although the centralized cooperative caching policies can obtain optimal caching decisions according to the global network status, a centralized entity to monitor and manage in-network caching is not possible [18], [26]. Specially, centralized cooperative caching policies are very difficult to implement in a large scale network. In contrast caching decisions are easily implemented at individual nodes independently that employ distributed cooperative caching policies [11]–[14] and [28]–[32]. However, employing distributed cooperative caching policies makes it difficult to control and manage caching nodes.

In [28], the authors proposed the WAVE caching policy, in which the upstream nodes utilize the chunk marking window to recommend the caching decisions for the downstream nodes. As the popularity of content increases, this window can exhibit exponentially growth. Thus, the popular contents are widely distributed and rapidly spread over the network. However, this policy can incur extra communication cost, which can increase as the number of cached content chunks increases. Wang *et al.* in [29] propose CPHR, in which allocate content partitions to corresponding node based on hash functions. Although this policy can reduce content redundancy and filer effects, it incurs expensive computational overhead. Guo *et al.* in [11] and Wang *et al.* in [30] proposed light-weight cooperative caching policies, in which different contents are distributed along the contents delivery paths according to the popularity of the contents. These proposed light-weight cooperative caching policies need to add an additional base at the nodes to maintain the caching states of the other cooperative nodes. Although the proposed policies in [11] and [30] can reduce the communication overhead and content redundancy, they thoroughly eliminate content redundancy, which leads to the costly transmission among caches, such as the approach in [31]. Ming *et al.* in [32] proposed an Age-based cooperative caching policy, in which the content cached at the nodes are replaced based on the lifetime. The lifetime of a content object is decided based on its location and popularity. Thus, the more popular a content is, the longer lifetime it has. However, the definition of lifetime for a content requires precise design, otherwise it will affect content replacement and network performance. In general, the existing cooperative caching policies still suffer from the following disadvantages. First, they can not make a trade-off between the hit ratio in the whole network and the local hit ratio. Although the existing policies can reduce the content redundancy and improve the hit ratio in the whole network, they incur costly transmission among caches due to the low local hit ratio. Second, they can not make a trade-off between the overhead and performance. Although the existing cooperative policies can reduce the access delay and improve the user experience, they incur a high cooperative overhead.

Therefore, to overcome the above disadvantages, we make the following observations concerning on cooperative caching algorithm:

1) **Classing the CCN topology:** To make the caching policies suitable for a large scale network, it is necessary to combine the centralized and distributed model by classing the CCN topology, which has a great impact on the network performance since there is a tradeoff between the overhead (e.g., communication and computation overhead) and performance (e.g., access delay).
2) **Eliminating content redundancy appropriately:** In CCN, content redundancy elimination is considered as one of the effective measures for improving caching performance. However, existing policies completely eliminate content redundancy, thus incurring costly transmission among caches. So the content redundancy should be eliminated appropriately.

Based on the above, we propose a cooperative caching algorithm in the two-level CCN, which is aimed at maximizing the provider earned profit.

3 System Model

In this section, we exhibit a new system model of the two-level CCN, including the network model and profit model.

3.1 Network Model

The two-level CCN network topology consists of control nodes and common nodes (i.e., black nodes and grey nodes in Fig. 1), where one control node and its neighboring common nodes form a SBS [10]. Besides caching, the control node, likes a logical control plane in [33], manages all common nodes by the index in the associated SBS and cooperates with other SBSs. However, the common node only caches contents and forwards requests. The copies of contents are cached at the control nodes along the contents delivery paths, while the common nodes only cache the contents discarded by its associated control node according to the Least Frequently Used (LFU) policy. Every control node maintains an index table to record which common node caches the discarded contents. The topology is centralized in each SBS and distributed in the whole network.

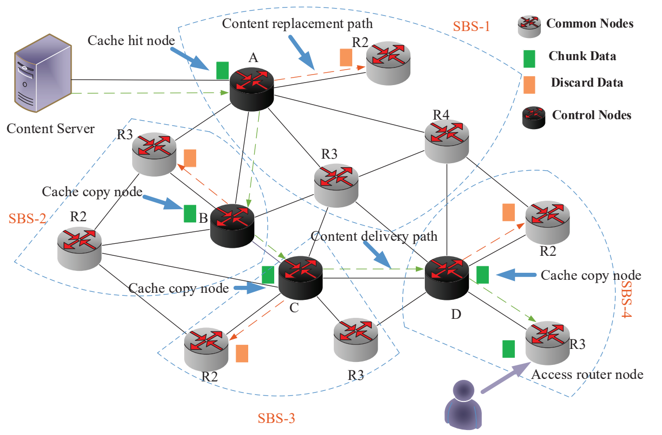

Fig. 1: The content caching and replacement scenario in CCN with multiple SBSs. This default caching strategy (i.e., the greedy non-collaboration strategy) can cause low content diversity and high content redundance within the whole network.

Therefore, it combines the advantages of the centralized structure with straight forward control and distributed structure with a certain scalability. The detailed process of constructing the two-level CCN is as follows:

- **Step 1:** Initially mark all nodes white.
- **Step 2:** Select the node of the largest degree and then color it black.
- **Step 3:** Color all adjacent nodes of the black node grey.
- **Step 4:** Select a gray node which has the largest white neighbor nodes, then we color it black and all its adjacent white nodes grey.
- **Step 5:** Repeat Step 3 until all the nodes are colored black or grey.

For ease of illustration, we assume that there are N control nodes in the network. Then the whole network consists of N SBSs $s_1, s_2, \ldots, s_N$. The contents requested by a user come from a finite library $F = \{f_1, \ldots, f_m, \ldots, f_M\}$ [14], whose size is equal to M, where f_m denotes the mth popular content. All contents of the library have the same size which is set to 1. Each control node can store K contents, while each common node can store H contents. The caching size of each SBS is $C_i, i \in \mathcal{N} = \{1, \ldots, N\}$. Thus, the whole network can cache $\sum_{i=1}^{N} C_i$ contents. Let $\mathcal{C}_i$ denote the set of contents cached at the SBS s_i. We assume that the request pattern is homogenous, then all nodes maintain the same content popularity distribution and obey the Zipf law, which is used to shape video popularity [34]. Accordingly, the popularity of the mth content from F is given by

$$p_m = \left(m^{\alpha} \sum_{m=1}^{M} m^{-\alpha} \right)^{-1}, \tag{1}$$

where α $(0 \leq \alpha \leq 1)$ denotes the Zipf parameter. As α increases, the requests are more focused on the popular contents. As m increases, the popularity of contents decreases, i.e., $p_1 > p_2 > \cdots > p_M$. The variable p_m also represents the probability of generating a request for content f_m in the whole network.

Since the control node maintains an index which records the caching information of the associated common nodes, both the caching information and routing table are considered in the routing process. Therefore, the detailed content search process in the two-level CCN is as follows: Firstly, as receiving a request, the node checks the local content store (CS). If the node caches the corresponding content, it directly forwards the content to the user. Otherwise, the type of this node needs to be determined. If it is a control node, it searches the index table whether there is a corresponding content cached at the associated common node. If it is a common node, the request is forwarded to the control node.Secondly, if the search in index table succeeds, the request is forwarded to the corresponding common node. Otherwise, the SBS cannot respond to the request. Then, the request is forwarded to any other SBS according to the routing table.Finally, due to the finite cache space, it is impossible to cache all kinds of contents in the network. Therefore, if the requested content is not cached in the network, we forward this request to the server according to the routing table.

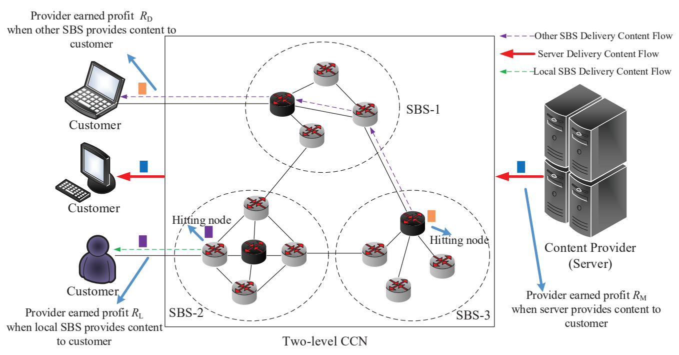
Provider earned profit R_D when other SBS provides content to customer
Other SBS Delivery Content Flow
Server Delivery Content Flow
Local SBS Delivery Content Flow
Customer
SBS-1
Customer
Hitting node
Hitting node
Content Provider (Server)
Customer
Provider earned profit R_L when local SBS provides content to customer
SBS-2
SBS-3
Provider earned profit R_M when server provides content to customer
Two-level CCN

Fig. 2: Content delivery flow and profit model.

3.2 Profit Model

Responding a request at the local SBS, at other SBSs, or the server will result in different transmission cost for the content provider (see Fig. 2). Meanwhile, as the transmission distance of the requested content increases (thereby resulting in longer transmission delay), the consumer will use the service provided by the provider less often, and then the lower profit is earned by the provider. Similarly, as the transmission distance of the requested content decreases (thereby resulting in shorter transmission delay), the consumer will use the service provided by the provider more often, and then the higher profit is earned by the provider.

Since the control node is adjacent to the common nodes in a SBS, we assume that their accessing cost for a content is approximately equal. Formally, let R_L denote the average earned profit that the request is served at the local SBS. We use R_D to denote the average earned profit that the request is served at other SBSs, and R_M denotes the average earned profit that the request is served at the server. In the given network structure, we can get $R_M < R_D < R_L$ [15].

In a practical scenario, provider sets the parameters R_L, R_D, and R_M based on the operational cost, such as energy consumption, bandwidth and transmission delay [15], [18]. The lower operational cost of provisioning content, the higher profit will be set by the provider.

4 Problem Formulation

Since the control node is adjacent to the common nodes in a SBS, we assume that their accessing cost for a content is approximately equal. For clarity, let $b_m^{(i)}$ $(i \in \mathcal{N}, m \in \{1, \ldots, M\})$ be the 0-1 content placement decision variable that indicates whether to cache content f_m in SBS s_i or not. For a content request in the network, the variables $p_m^{(c)}$, $p_m^{(r)}$ and $p_m^{(i)}$ denote the probabilities of generating a request for content f_m in a common node c, a control node r and the associated SBS s_i, respectively. We define n_i $(i \in \mathcal{N})$ as the number of common nodes in a SBS s_i. All nodes have the same request probability for a same content in the homogenous request model, then $p_m^{(i)} = \sum_{c=1}^{n_i} p_m^{(c)} + p_m^{(r)} = (n_i + 1)p_m^{(c)} = (n_i + 1)p_m^{(r)}$. Thus, the popularity of the mth popular content is

$$p_m = \sum\nolimits_{i=1}^{N} p_m^{(i)} = \sum\nolimits_{i=1}^{N} (n_i + 1) p_m^{(r)}. \tag{2}$$

4.1 Twofold Benchmark Problem

Firstly, we solve the problem of maximizing the local hit ratio in the SBS as follows

$$\max_{\left\{b_m^{(i)}\right\}_{i=1}^{N}} F_1 = \sum_{i=1}^{N}\sum_{m=1}^{M} b_m^{(i)} \cdot p_m^{(i)}, \tag{3}$$

$$s.t. \sum_{m=1}^{M} b_m^{(i)} \leq C_i, \forall i \in \{1,\ldots,N\}.$$

According to (3), if each SBS caches the most popular contents independently, it can reach the maximum of the local hit ratio. Due to employing the greedy non-collaboration strategy [12], [10], the control node of each SBS caches the same set of contents, which results in the low content diversity and provider profit.

Secondly, we solve the problem of maximizing the hit ratio in the whole network

$$\max_{C} F_2 = \sum_{m=1}^{C} B_m \cdot p_m, \tag{4}$$

where C denotes the available cache size (i.e., the number of content items that can be cached) in the whole network, and B_m ($m \in \{1,\ldots,M\}$) is a binary function. If the content f_m is cached, $B_m = 1$; otherwise, $B_m = 0$. If collaboration among SBSs does not exist, we have $C = C_{\max}$ where $C_{\max} \geq C_i$ ($i \in \mathcal{N}$). If collaboration among SBSs exists, i.e., each SBS caches a content set with different types, we have $C = \sum_{i=1}^{N} C_i$. Then we can easily get $C_{\max} < \sum_{i=1}^{N} C_i$.

According to (4), if the same contents are not cached in different SBSs, it can reach the maximum of the local hit ratio within the whole network. The full collaboration for all N SBSs [35] results in greater content diversity in the whole network. Then many requests can be satisfied in the network without forwarding to the server. However, it leads to frequent access among SBSs. The profit owing to hitting content at other SBSs is lower than that of hitting at a SBS locally.

4.2 Maximization of the Average Earned Profit by Provider

The optimal objective of maximizing the average earned profit is

$$\max_{\left\{b_m^{(i)}\right\}_{i=1}^{N}} R_{\text{avg}} = R_L \sum_{i\in\mathcal{N}} H_i^i + R_D \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}, j\neq i} H_i^j + R_M \sum_{i\in\mathcal{N}} H_i^{N+1}, \tag{5}$$

$$s.t. \sum_{m=1}^{M} b_m^{(i)} \leq C_i, \forall i \in \{1,\ldots,N\},$$

where

$$H_i^i = \sum_{m=1}^{M} b_m^{(i)} \cdot p_m^{(i)}, \tag{6}$$

$$H_i^j = \sum_{m=1}^{M} b_m^{(j)} \cdot p_m^{(i)}, \tag{7}$$

$$H_i^{N+1} = \sum_{m=1}^{M} (1 - B_m) \cdot p_m^{(i)}. \tag{8}$$

The variable H_i^i denotes the local hit ratio that a request from SBS s_i is responded at local SBS. Let H_i^j denote the remote hit ratio that a request from SBS s_i is responded at SBS $s_j (j \in \mathcal{N})$. Denoting the variable H_i^{N+1} as the miss ratio that a request from SBS s_i is responded to at the server.

The above problem is a well-known 0-1 multi-knapsack problem which is NP-complete [15] – [17]. It is difficult to determine the optimal $b_m^{(i)}$ for $i \in \{1,\ldots,N\}$ and $m \in \{1,\ldots,M\}$, especially when there is a large collection of SBSs and contents. However, based on the twofold benchmark problem, we can get some guiding principles on how to effectively increase the earned profit.

5 A Heuristic Cooperative Caching Algorithm

According to the above analysis, the optimal objective of maximizing the average earned profit is converted into an optimal content placement problem. To solve this problem, a heuristic cooperative caching algorithm is proposed. Meanwhile, we also propose a replacement policy to support the caching algorithm.

5.1 Optimal Content Placement

In this subsection, we analyze an optimal content placement problem that is aimed at maximizing average provider earned profit by making a trade-off between the hit ratio in a SBS and the hit ratio in the whole network. For ease of illustration, let $P_L = \sum_{i\in\mathcal{N}} H_i^i$, $P_D = \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N},j\neq i} H_i^j$ and $P_M = \sum_{i\in\mathcal{N}} H_i^{N+1}$ denote the local SBSs hit ratio, the other SBSs hit ratio and the server hit ratio, respectively. The above equation (5) can be simplified as

$$R_{\text{avg}} = P_L R_L + P_D R_D + P_M R_M, \tag{9}$$

where

$$P_M = 1 - P_L - P_D. \tag{10}$$

Since, the parameter R_L and R_D are the given constants, let $R_D = \tau R_L$, where $0 \leq \tau \leq 1$. As time goes on, the amount of contents that are cached at the small cells increases. The requests do not need to be forwarded to the server leading to a smaller value of R_M, then we let $R_M = 0$. The above equation (9) can be rewritten as

$$R_{\text{avg}} = (P_L + P_D\tau)R_L. \tag{11}$$

The probability of finding the content in SBS s_i can be written as $P_L^i = \sum_{m\in\mathcal{C}_i} p_m$, and thus the probability of finding a content item at any given SBS in the network is $\sum_{i=1}^{N} P_L^i/N$ or $\sum_{i=1}^{N}\sum_{m\in\mathcal{C}_i} p_m/N$. This is also the average local hit ratio H_L, which can be simplified as

$$P_L = \frac{1}{N}\sum_{m=1}^{M} d_m p_m, \tag{12}$$

where d_m denotes the number of duplications of content f_m within the network. Since the maximum number of content items cached within the network is $C = \sum_{i=1}^{N} C_i$, the parameter M in (12) can be replaced by C.

Let B denote the set of all cached contents in the network. The probability of finding a content in the network can be written as $\sum_{m\in B} p_m$. The variable $\sum_{m\in B} p_m$ denotes the hit ratio in the whole network which is equal to $1 - P_M$. Expressing $\sum_{m\in B} p_m$ as $1 - P_M$ and substituting the value of P_L from (12) in (10), we can obtain $P_D = \sum_{m\in B} p_m - \frac{1}{N}\sum_{m=1}^{C} d_m p_m$. Substituting the expression for P_L and P_D into (11), we can obtain

$$R_{\text{avg}} = \left((1-\tau)\frac{1}{N}\sum_{m=1}^{C} d_m p_m + \tau \sum_{m\in B} p_m \right) R_L. \tag{13}$$

For a given τ, the average profit in (13) is a function of the vector $\vec{d} = (d_1, d_2, \ldots, d_M)$. A content placement $\vec{d}$ is optimal when it incurs maximum average provider earned profit in (13).

Lemma 1. *For any popularity-based content request pattern, such as the Zipf distribution, the optimal placement scheme must obey the following constraint in a steady state.*

A content should not be cached in the network when at least one more popular content is not cached in the network. In other words, the content f_m (i.e., m-th popular content) can not be cached when a higher popularity content f_k ($k < m$) is not cached. This is the *popularity cache constraint*.

Proof. Employing the proofs by contradiction, we assume that there exists an optimal placement that maximizes the average provider earned profit in (13) and violates the *popularity cache constraint*. This means that a content f_m does not exist in the network (i.e., $d_m = 0$) while a less popular content f_l exists in the network (i.e., $l > m, n_l > 0$). ■

According to the equation (13), it can be observed that if content f_l is replaced with a more popular content f_m, the profit will be higher. This contradicts our assumption, and thus the optimal content placement must obey the *popularity cache constraint*.

Now, we assume that the parameter f_T is the least popular content which is cached in the network. Based on the *popularity cache constraint*, there is at least one duplication of contents from f_1 to f_T in the network. Therefore, the (13) can be rewritten as

$$R_{\text{avg}} = \left((1-\tau)\frac{1}{N}\sum_{m=1}^{T} d_m p_m + \tau \sum_{m=1}^{T} p_m \right) R_L. \quad (14)$$

Lemma 2. In the optimal content placement, a content f_k (i.e., kth popular content) should not be copied unless all other more popular contents have been copied in all SBSs.

Proof. According to the above constraint, there exists at least one duplication of content from f_1 to f_T in the network. Therefore, the (13) can be rewritten as

$$R_{\text{avg}} = \left(\sum_{m=1}^{T} \left((1-\tau)\frac{1}{N} d_m + \tau \right) p_m \right) R_L. \quad (15)$$

Now we let $d_\ell \neq d_k$, $1 < d_\ell, d_k < N$, and $\ell < k$. It can be seen that a higher profit is possible by increasing the d_ℓ while decreasing the d_k. This case can imply the following claim: when there is space for caching more duplications of content f_ℓ (i.e., $d_\ell < N$), less popular contents (e.g., content f_k, $k > \ell$) should not be copied. Obeying the above claim, we can not copy the content f_2 unless we have the duplications of content f_1 in all SBSs (i.e., $d_1 = N$). So, we can not copy the content f_m unless we already have the duplications of more popular contents in all SBSs. ■

Claim. The optimal content placement $\vec{d}$ has the following characteristics:

1)$n_m = N$ for $1 \le m \le \ell$, where ℓ denotes the least popular content that are cached in the network. The value of ℓ should be determined by τ. There is one duplication of contents from f_1 to f_ℓ that will be cached in all SBSs.

2)$d_m = 1$ for $\ell + 1 \le m \le T$, where $T = \sum_{i=1}^{N} C_i - N\ell + \ell + 1$. This means that the remaining cache space will cache the unique contents.

3)d_m=0 for $m > T$.

Proof. Based on the *Lemma 1*, there must exists at least one duplication of contents form f_1 to f_T in the network. *Lemma 2* illustrates that a content should not be copied before all other more popular contents have been copied in all SBSs. This means that if ℓ denotes the least popular content which are cached in the network, there should exist N duplications of contents from f_1 to f_ℓ in the network. ■

Note that the value of ℓ is not specified in the above analysis, and actually we do not know how many contents should be copied in all SBSs to achieve the optimal content placement. It only describes that if the optimal solution needs the copy, it must be across all SBSs. In the following subsection we will illustrate how to get the value of ℓ.

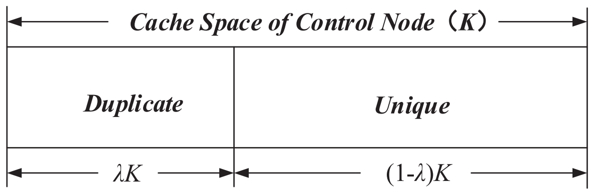

Fig. 3: Control node cache partitioning in a heuristic caching algorithm.

5.2 *A Heuristic Caching Algorithm*

In order to realize the optimal content placement, an optimal heuristic cooperative caching algorithm is proposed. The algorithm divides the cache space of control nodes into two fractions by a split factor λ ($0 \le \lambda \le 1$) in a SBS (see Fig.3). One is a duplicate fraction (λK portion) to cache the same copies of the most popular contents. The other is a unique fraction ($(1-\lambda)K$ portion) that only caches the different contents which are not cached at any other control nodes. With the caching policy of common nodes, they only store the contents discarded by duplicate fraction according to control node replacement policy. Therefore, a SBS caches the contents which are also ever cached at its control node. This proposed caching algorithm makes it possible to increase the survival time of the most popular contents so as to enhance the hit ratio[32].

- In order to increase the earned profit on serving requests at the local SBS, there should be a duplicate fraction in every SBS cache space, i.e., $Z_\lambda(\lambda K + n_i H)$, to cache the most popular contents. The minimum duplicate fraction is $Z_\lambda(\lambda K + n_{\min} H)$ for each SBS, where $n_{\min}$ denotes the minimum number of common nodes for each SBS in the network, and Z_λ denotes that the binary function is equal to 1 when $0 < \lambda \le 1$ and 0 otherwise.
- In order to increase the earned profit on serving request in the whole network, there should be a unique fraction in every SBS cache space, i.e., $(1-\lambda)K$. The size of this fraction is equal to the size of the unique fraction of the control node caching space.

At steady state, the total number of different contents cached in the network is $Z_\lambda(\lambda K + n_{\max} H) + N(1-\lambda)K$, where $n_{\max}$ denotes the maximum number of common nodes for each SBS in the network.

5.3 *Control Node Cache Replacement*

Algorithm 1 Control node cache replacement policy

```
INPUTS: A data packet $O_{new}$;
BEGIN: Checking the label of the packet header;
  if Labeling value is equal to one then
      $O_{\min}$ = the least popular content in the duplicate fraction;
  else
      $O_{\min}$ = the least popular content in the whole cache space;
  end if
  if $O_{new}.popularity > O_{\min}.popularity$ then
      Replacing $O_{new}$ with $O_{\min}$;
      if Replaced content which comes from the duplicate fraction then
          Caching $O_{\min}$ at the associated common node;
      else
          Discarding the $O_{\min}$;
      end if
  else
      Keeping original content;
  end if
```

For the control node replacement policy, the first fraction of the control node caching space caches *duplicated* contents downloaded from the other SBS rather than the server. In the second fraction, the control node only caches the *unique* contents that need to be downloaded from the server. For distinguishing the type of content, the data packet header uses a label (only one bit) to mark the source of contents. For caching a new *unique* content whose label is 0, the least popular content in the whole cache will be replaced when cache space is full. For caching a *duplicated* content whose label is 1, the discarded content is only selected from the first duplicate fraction of the cache space. In this case, the evicted content will be cached at the associated common node. The pseudocode of control node cache replacement policy is shown in Algorithm 1.

5.4 *Average Provider Earned Profit*

Due to $R_M = 0$, the equation (5) can be simplified as

$$\max_{b_m^{(i)}} R_{\text{avg}} = R_L \sum_{i\in\mathcal{N}} H_i^i + R_D \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}, j\neq i} H_i^j. \tag{16}$$

To compute the average provider earned profit, we should know H_i^i and H_i^j used in (16). The local SBS s_i hit ratio consists of the duplicate fraction hit ratio H_{id}^i and the unique fraction hit ratio H_{iu}^i. Furthermore, the probability that the request from SBS s_i is served at unique fraction of other SBS s_j is H_{iu}^j. For $N(1-\lambda)K$ *unique* contents, we assume that these *unique* contents are uniformly cached at each unique fraction. Those can be expressed as

$$H_{id}^i = \sum_{m=1}^{\Lambda_i} p_m^{(i)}, \tag{17}$$

$$H_{iu}^i = \frac{\sum_{m=\Lambda_{\max}+1}^{\Gamma} p_m^{(i)}}{N}, \tag{18}$$

$$H_{iu}^j = \frac{\sum_{m=\Lambda_{\max}+1}^{\Gamma} p_m^{(i)}}{N}, i \neq j, \tag{19}$$

where $\Gamma = \Lambda_{\max} + N(1-\lambda)K, \Lambda_i = Z_\lambda(\lambda K + n_i H)$, and $\Lambda_{\max} = Z_\lambda(\lambda K + n_{\max} H)$. The local SBS s_i hit ratio is $H_i^i = H_{id}^i + H_{iu}^i$, the hit ratio at other SBS s_j rather than the associated one is $H_i^j = H_{iu}^j + \sum_{i=\Lambda_i+1}^{\Lambda_j} p_m^{(i)}$, and the miss ratio is $H_i^{N+1} = 1 - H_i^i - (N-1)H_i^j$, respectively.

Substituting the expressions of H_i^i and H_i^j into (16), we convert the problem of maximizing average earned profit to the problem of determining the optimal split factor λ_{opt}, i.e., $\max_\lambda R_{\text{avg}}$, where

$$\begin{aligned} R_{\text{avg}} = R_L &\left(\sum_{i=1}^{N} \sum_{m=1}^{\Lambda_i} p_m^{(i)} + \frac{\sum_{i=1}^{N} \sum_{m=\Lambda_{\max}+1}^{\Gamma} p_m^{(i)}}{N} \right) \\ + R_D &\left(\frac{(N-1) \sum_{i=1}^{N} \sum_{m=\Lambda_{\max}+1}^{\Gamma} p_m^{(i)}}{N} \right. \\ &\left. + \sum_{i=1}^{N} \sum_{j\in\mathcal{N}, j\neq i} \sum_{m=\Lambda_i+1}^{\Lambda_j} p_m^{(i)} \right). \end{aligned} \tag{20}$$

Substituting (2) into (20), we have

$$\begin{aligned} R_{\text{avg}} &= R_L \sum_{i=1}^{N} \beta_i \sum_{m=1}^{\Lambda_i} p_m \\ &+ \sum_{i=1}^{N} \beta_i \sum_{m=\Lambda_{\max}+1}^{\Gamma} p_m \left(\frac{R_L}{N} + \frac{(N-1)R_D}{N} \right) \\ &+ R_D \sum_{i=1}^{N} \beta_i \sum_{j\in\mathcal{N}, j\neq i} \sum_{m=\Lambda_i+1}^{\Lambda_j} p_m, \end{aligned} \tag{21}$$

where $\beta_i = \frac{n_i+1}{\sum_{i=1}^{N} n_i+1}$ denotes the proportion of the number of nodes in SBS s_i to the number of all nodes in the network. From (21), we have

$$\begin{aligned} \max_\lambda R_{\text{avg}} &= R_L \sum_{i=1}^{N} \beta_i \frac{\Lambda_i^{1-\alpha} - 1}{M^{1-\alpha} - 1} \\ &+ \sum_{i=1}^{N} \beta_i \left(\frac{\Gamma^{1-\alpha} - 1}{M^{1-\alpha} - 1} - \frac{(\Lambda_{\max}+1)^{1-\alpha} - 1}{M^{1-\alpha} - 1} \right) \\ &\left(\frac{R_L}{N} + \frac{(N-1)R_D}{N} \right) \\ &+ R_D \sum_{i=1}^{N} \beta_i \sum_{j\in\mathcal{N}, j\neq i} \left(\frac{\Lambda_j^{1-\alpha} - 1}{M^{1-\alpha} - 1} - \frac{(\Lambda_i+1)^{1-\alpha} - 1}{M^{1-\alpha} - 1} \right), \end{aligned} \tag{22}$$

where the equality follows from the relation $\sum_{m=1}^{k} p_m \approx \frac{k^{1-\alpha}-1}{M^{1-\alpha}-1}$ [12]. By equating the derivation of the above equation to zero, we can compute the λ_{opt} at which profit is maximized.

6 Simulation Results

The performance of the proposed heuristic cooperative caching algorithm was evaluated using the previously derived equations in Section 5, and then via open source ndnSIM [36] network simulation (version 2.1), which can provide basic structure of the CCN node, i.e., policy layer, Content Store (CS), Pending Interest Table (PIT) and Forwarding Information Base (FIB). There are 4 SBSs in the network topology, where the backbone consists of 4 control nodes as shown in Fig.4.

For simulation, the arrival of the requests follows a Poisson process and the content popularity distribution obeys the Zipf law. To support the proposed caching algorithm in the ndnSIM, we

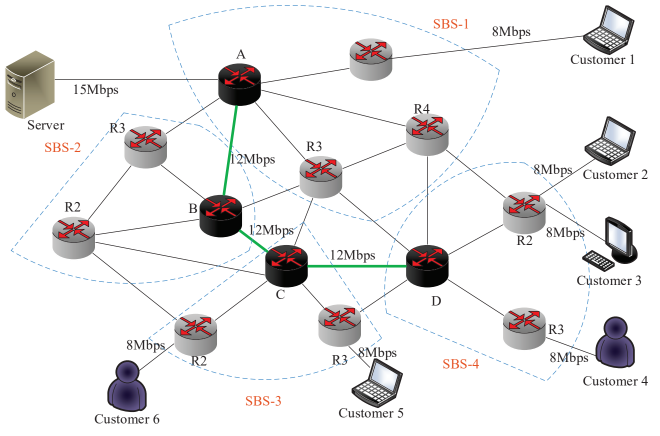

Fig. 4: Simulation topology.

install a cache module named *caching-duplicated-unique* on the control node, which enable the split CS implement different caching schemes. The duplicate fraction caches the same copies of the most popular contents, while the unique fraction caches the different contents which cannot be cached at any other control nodes. We also add a tag module named *content-source-tag* in the data packet to support our replacing policy of control node in the ndnSIM. When the value of tag is one, the coming content belongs to the *duplicated* content. When the value of tag is zero, the coming content belongs to the *unique* content. In addition, we perform simulation 20 times, and each time is 500s. We acquire the average values as the simulation results. For illustrative convenience, let $R_D = \tau R_L$ where $0 \leq \tau \leq 1$, and $\gamma = \frac{K}{M}$ which denotes the cache-library ratio. The simulation runs with the setting benchmark parameters, which are shown in Table 1.

Table 1 Baseline Simulation Parameters

Parameters	**Values**
Number of SBSs in the whole network (N)	4
Number of common nodes in each SBS (n_i)	3,2,2,2
Cache size in each common node (H)	20
Earned profit (R_L,R_M)	20,0
Rebate-to-earn-profit ratio (τ)	$0 \leq \tau \leq 1$
cache-library ratio (γ)	$0 \leq \gamma \leq 1$
The size of content library (M)	1000

6.1 Hit Ratio and Earned Profit

Fig. 5(a) describes the effects of split factor λ on the hit ratios when $\alpha = 0.8$ and $\gamma = 0.05$. In the case of $\lambda = 0$, control node of each SBS will cache different contents and common nodes of each SBS do not cache any content, which leads to frequent communication among SBSs and can not effectively use caching resource of common nodes. Therefore, when the value of λ increases a little (i.e., λ=0.1), the local hit ratio (P_L) will increase rapidly, while the remote hit ratio (P_D) will decrease rapidly. In the case of $\lambda = 1$, the unique fraction of the control node is zero, which leads to a low content diversity, and then $P_D = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}, j \neq i} \sum_{m=\Lambda_i+1}^{\Lambda_j} p_m^{(i)}$. A large λ value will result in more copies of the popular contents cached at the local SBS, which brings high local SBS hit ratio (P_L) and low remote hit ratio (P_D). The larger λ value is, the easier the requested content can be found.

The probability that the request is hit on the server P_M is related to the total number of the *unique* contents in the network. When $\lambda > 0.1$, P_M will increase with the increase of λ, which means there is a lower content diversity. However, when $0 \leq \lambda \leq 0.1$, the common nodes gradual begin to cache the content. In this case, the diversity of content will increase. From Fig. 5(a), the difference between the theoretical value and the simulation value is very small, indicating that the equations derived in Section 5 are correct.

Fig. 5(b) and Fig. 5(c) depict the relation between earned profit and λ when $\alpha = 0.8$ and $\gamma = 0.05$. When $\tau = 0.2$ and $\tau = 0.7$, the profit will increase with the increase of λ; but after $\lambda \geq \lambda_{\text{opt}}$, the profit will decrease with the increase of λ as shown in Fig. 5(b). This λ_{opt} can be derived by making equation (22) equal to 0. As long as λ is not equal to 0, λ_{opt} always exists, which is used to get the maximum profit. In other words, it converts searching for the maximum of earned profit to the determination of λ_{opt}. When $\tau = 0.2$ and $\tau = 0.7$, λ_{opt} is equal to 0.6 and 0.1 respectively. Thus, a smaller λ_{opt} is needed when local profit R_L and remote profit R_D are not much different.

From Fig. 5(c), when $\tau = 0$ (i.e., $R_D = 0$), the equation (11) can be simplified as $R_{\text{avg}} = R_L P_L$, which means profit is only related to the P_L for a given R_L, where P_L increases with the increase of λ. Therefore, when $\tau = 0$, $\lambda = 1$ gives rise to the maximum P_L, i.e., the maximum profit. When $\tau = 1$ (i.e., $R_L = R_D = 20$), the equation (11) can be written as $R_{\text{avg}} = R_L(1 - P_M)$, whose value is only related to P_M. This can be used to explain why the profit increases with the decrease of λ. Intuitively, when $R_L = R_D$, there is no benefit of caching content in local SBSs. In this case, the only measure to increase profit is to find minimum P_M.

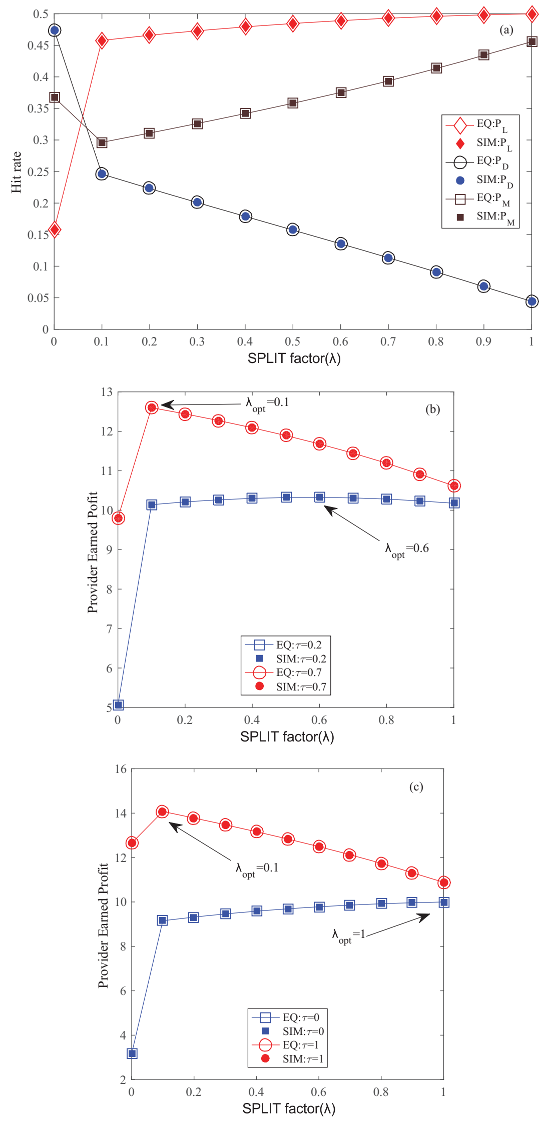

Fig. 5: (a) Hit Ratio vs. Split Factor (λ). (b) and (c) Provider Earned Profit vs. Split Factor (λ) when $\tau = 0, 0.2, 0.7, 1$.

Fig. 6 and Fig. 7 describes the impacts of the Zipf parameter α and the cache-library ratio γ on the average provider earned profit R_{avg}. Due to the feature of Zipf parameter, it is easily seen that a higher R_{avg} is achieved with a greater α. When the requests from the users become more focused on the popular contents, the local SBS can respond to the more requests. Hence, the providers can earn more profit. Likewise, as γ increases, because the SBS can cache more contents that increases the local hit ratio, the provider earned profit can be effectively increased. However, the value of γ is irrelevant to the optimal λ. Additionally, we can clearly seen that the optimal split factor λ increases with a greater α from Fig. 6. The reason is that the requests from the users become more focused on the popular contents, the bigger duplicate fraction is needed when the value of τ is small. On the other hand, from Fig. 7, we will find the optimal λ remains the same with an increase in α when the value of τ is large. The reason is that a smaller λ_{opt} is needed when the local profit R_L and remote profit R_D are not much different. Furthermore, with a greater τ, a clearly higher profit is achieved.

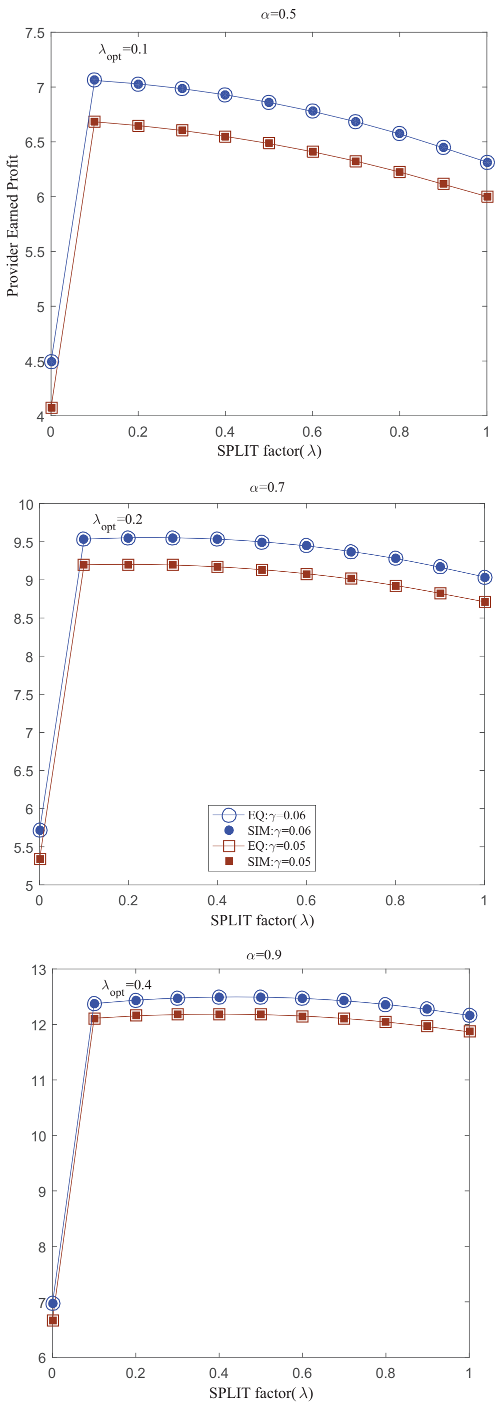

Fig. 6: The impacts of α and γ on the average provider earned profit when $\tau = 0.3$.

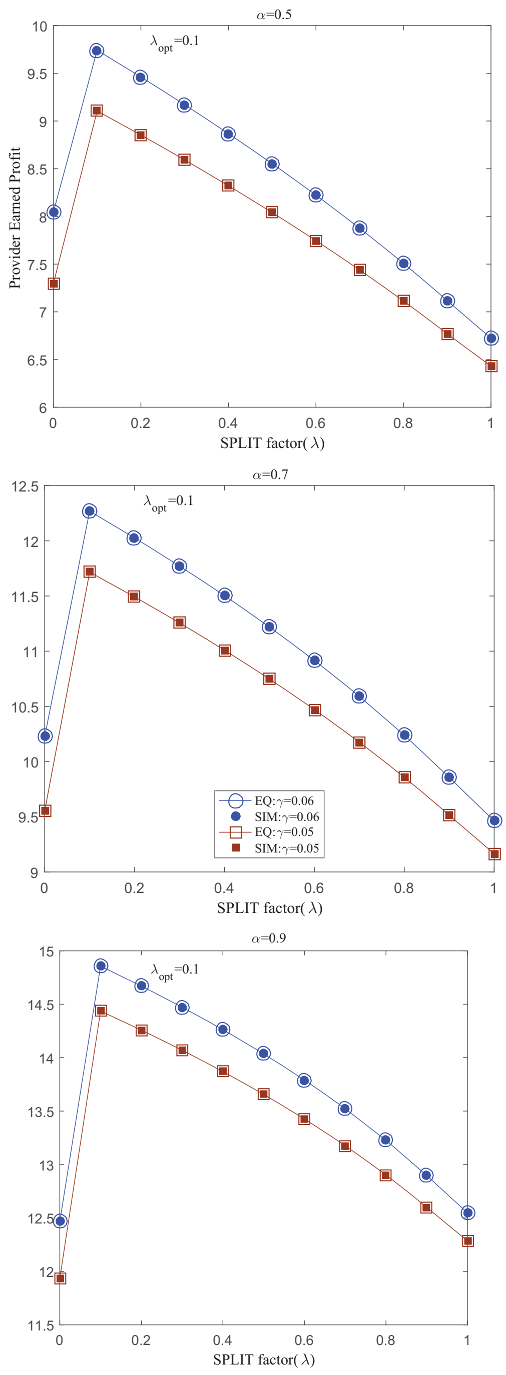

Fig. 7: The impacts of α and γ on the average provider earned profit when $\tau = 0.8$.

6.2 *Comparison with Traditional Caching Strategies*

Recently, some caching algorithms are proposed for solving wireless caching problems. Because of the difference between the wireless communication and wired communication, the wireless caching schemes cannot be directly applied to a structured wired environment, while the traditional caching strategies can be applied to wireless and wired networks. Therefore, in this subsection, we use some representative traditional caching strategies such as Least Recently Used (LRU), Random (RNDM) and Least Frequently Used (LFU) in [37] as performance benchmarks. In Fig. 8, let $\alpha = 0.8$ and $\gamma = 0.05$, and then we set λ to 0, 1, λ_{opt}. Among them, LRU and LFU cache content based on content popularity, while RNDM does not. As expected, when $\lambda = \lambda_{\text{opt}}$, the profit reaches maximum as shown in Fig. 5(b) and 5(c). When $\lambda = 0$, since the impact of P_D on the profit will increase from (11), the profit line is slowly approaches to the optimal curve with τ increasing. However, it can never can reach close to the best performance, because the cache space of common nodes can not be fully used when $\lambda = 0$.

From Fig. 8, when $\lambda = 1$, since the impact of P_L on the profit will increase from (11), and the profit line is very close to the optimal curve as the value of τ decreases. P_L reaches maximum when $\lambda = 1$. The profit lines of all traditional strategies are located between lines of our algorithm with $\lambda = 1$ and $\lambda = 0$ as shown in Fig. 8. Since the RNDM caches contents randomly, its performance is superior to the other two traditional caching strategies when τ is larger than a certain value. Since LFU is very sensitive to popularity, its profit line is very close to the optimal curve when $\lambda = 1$.

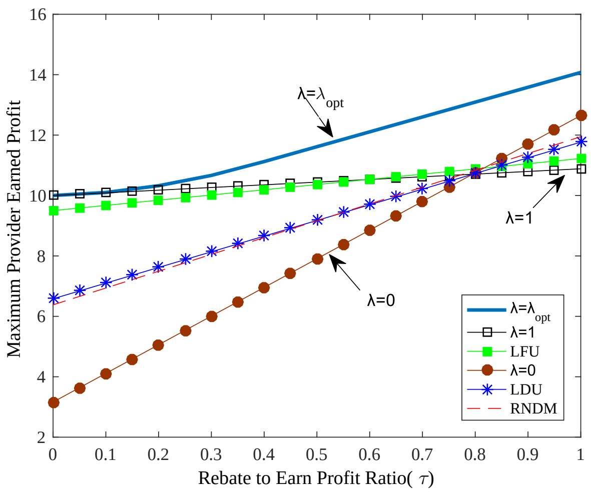

Fig. 8: Comparison with some traditional strategies.

7 Conclusion

In this work, in order to avoid the disadvantages caused by the CCN with arbitrary topology and to achieve the goal of maximizing the earned profit, we propose a heuristic collaborative cache algorithm under a CCN two-level topology to achieve optimal content placement. In this algorithm, the cache space of control node is divided into two fractions. One fraction caches the most popular contents which could be cached in any control node, while the other caches the less popular contents. We also propose a caching replacement policy for the control nodes corresponding to our collaborative cache algorithm. The optimal split factor has been derived and verified in NDNsim simulations. Finally, we compare our algorithm with some traditional caching strategies to verify its performance, and the results indicate that our algorithm achieves significant profit gains.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants no. U1604155, no. 61871430, no.61602155, and U1404611, and in part by the Scientific and Technological Innovation Team of Colleges and Universities in Henan Province under Grants No.20IRTSTHN018, and in part by the basic research projects in the University of Henan Province under Grants No. 19zx010, and in part by the Science and Technology Development Programs of Henan Province under Grant No. 192102210284.

8 References

1 Dai J., Hu Z., Li B., Liu J.: 'Collaborative hierarchical caching with dynamic request routing for massive content distribution'. Proc. IEEE International Conference on Computer Communications (INFOCOM), Orlando, USA, Mar. 2012, pp. 2444–2452

2 Woo S., Jenog E., Park S., et al.: 'Comparison of caching strategies in modern cellular backhaul networks'. Proc. MobiSys'13 Proceeding of the 11th annual International Conference on Mobile Systems, Applications, and Services (MobiSys), Taiwan, China, Jun. 2013, pp. 319–332.

3 Ge A., Yang B., Ye J., et al.: 'Spatial spectrum and energy efficiency of random cellular networks', *IEEE Transactions on Communications*, 2015, **63**, (3), pp. 1019–1030

4 Kvaternik K., Llorca J., Kilper D., Pave L.: 'A Methodology for the Design of Self-Optimizing, Decentralized Content-Caching Strategies', *IEEE/ACM Transactions on Networking*, 2016, **24**, (5), pp. 2634–2647

5 Zhang X., Lv T., Ni W., Cioffi J., et al.: 'Energy-Efficient Caching for Scalable Videos in Heterogeneous Networks', *IEEE Journal on Selected Areas in Communications*, 2018, **36**, (8), pp. 1802–1815

6 Psaras I., Chai W. K., Pavlou G.: 'Probabilistic in-network caching for information-centric networks'. Proc. ACM SIGCOMM Workshop Information Centric Networking (ICN)'12, Helsinki, Finland, Aug. 2012, pp. 55–60

7 Zhang M., Xie P., Zhu J., et al.: 'NCPP-based caching and NUR-based resource allocation for information-centric networking', *Journal of Ambient Intelligence Humanized Computing*, 2019, **10**, (5), pp. 1739–1745

8 Chen B., Yang C., Wang G.: 'High Throughput Opportunistic Cooperative Device-to-Device Communications With Caching', *IEEE Transaction on Vehicular Technology*, 2017, **66**, (8), pp. 7527–7539

9 Song F., Ai Z., Li J., et al.: 'Smart Collaborative Caching for Information-Centric IoT in Fog Computing', *Sensors*, 2017, **17**, (11), pp. 2512

10 Xu Y., Li Y., Lin T., Zhang G.: 'A dominating-set-based collaborative caching with request routing in content centric networking'. Proc. IEEE International Conference on Communications (ICC), Budapest, Hungary, Jun. 2013, pp. 3624–3628

11 Guo S., Xie H., Shi G., 'Collaborative Forwarding and Caching in Content Centric Networks'. International Conference on Research in Networking, Heidelberg, Germany, May 2012, pp. 41–45

12 Taghizadeh M., Micinski K., Ofria C., et al.: 'Distributed cooperative caching in social wireless networks', *IEEE Transaction on Mobile Computing*, 2013, **12**, (6), pp. 1037–1053

13 Ahlehagh H., Dey S.: 'Video caching in radio access network: impact on delay and capacity'. Proc. IEEE Wireless Communications and Networking Conference (WCNC), Shanghai, China, Apr. 2012, pp. 1–6.

14 Chen Z., Lee J., Quek T. Q. S., Kountouris M.: 'Cooperative caching and transmission design in cluster-centric small cell networks', *IEEE Transaction on Wireless Communication*, 2017, **16**, (5), pp. 3401–3415

15 Abdallah K., Chakareski J.: 'Collaborative caching for multicell-coordinated systems'. Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Hong Kong, China, Apr. 2015, pp. 257–262

16 Shanmugam K., Golrezaei N., Dimakis A. G., et al.: 'FemtoCaching: wireless content delivery through distributed caching helpers', *IEEE Transaction on Information Theory*, 2013, **59**, (12), pp. 8402–8413

17 Martello S., Toth P.: 'Knapsack Problems: Algorithms and Computer Implementations' (JOHN WILEY & SONS, New York, 1990)

18 Li Q., Zhang C., Ge X., et al.: 'A cost-oriented cooperative caching for software-defined radio access networks'. Proc. IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Valencia, Spain, Sep. 2016, pp. 1–6.

19 Arslan M. Y., Sundaresan K., Rangarajan S.: 'Software-defined networking in cellular radio access networks: potential and challenges', *IEEE Communications Magazine*, 2015, **53**, (1), pp. 150–156

20 Kreutz D., Ramos F. M. V., Verssimo P. E., et al.: 'Software-defined networking: a comprehensive survey', *Proceedings of the IEEE*, 2015, **103**, (1), pp. 14–76.

21 Fan L., Cao P., Almeida J., et al.: 'Summary cache: a scalable wide-area web cache sharing protocol', *IEEE/ACM Transaction on Networking*, 2000, **8**, (3), pp. 281–293

22 Borst S., Gupta V., Walid A.: 'Distributed caching algorithms for content distribution networks'. Proc. IEEE International Conference on Computer Communications (INFOCOM), San Diego, USA, Mar. 2010, pp. 1478–1486.

23 Rhea S., Godfrey B., Karp B., et al.: 'OpenDHT: a public DHT service and its uses', *Acm Sigcomm Computer Communication Review*, 2005, **35**, (4), pp. 73–84

24 Wang S., Bi J., Wu J.: 'On performance of cache policy in information-centric networking'. Proc. 21st International Conference on Computer Communications and Networks (ICCCN), Munich, Germany, Aug. 2012, pp. 1–7

25 Williamson C.: 'On filter effects in web caching hierarchies', *Acm Transactions on Internet Technology*, 2002, **2**, (1), pp. 47–77

26 Li Y., Xie H., Wen Y., Zhang Z.: 'Coordinating in-network caching in content-centric networks: model and analysis'. Proc. IEEE International Conference on Distributed Computing Systems (ICDCS), Philadelphia, USA, Jul. 2013, pp. 62–72

27 Tran T., Le D., Yue G., Pompili D.: 'Cooperative Hierarchical Caching and Request Scheduling in a Cloud Radio Access Network', *IEEE Transactions on Mobile Computing*, 2018, **17**, (12), pp. 2729–2743

28 Cho K., Lee M., Park k., er al.: 'WAVE: Popularity-based and Collaborative In-network Caching for Content-Oriented Networks'. Proc. IEEE International Conference on Computer Communications (INFOCOM), Orlando, USA, Mar. 2012, pp. 316–321

29 Wang S., Bi J., Wu J., Vasilakos A. V.: 'CPHR: In-Network Caching for Information-Centric Networking With Partitioning and Hash-Routing', *IEEE/ACM Transactions on Networking*, 2016, **24**, (5), pp. 2742–2755

30 Wang X., Ren J., Tong T., et al.: 'Towards Efficient and Lightweight Collaborative In-Network Caching for Content Centric Networks'. Proc. IEEE Global Communications Conference (GLOBECOM), Washington, USA, Dec. 2016, pp. 1–7

31 Wang M., Zhang J., Bensaou B., 'Intra-AS Cooperative Caching for Content-Centric Networks'. Proc. of the 3rd ACM SIGCOMM workshop on Information-centric networking, Hong Kong, China, Aug. 2013, pp. 61–66

32 Ming Z., Xu M., Wang D., 'Age-based cooperative caching in information-centric networks'. Proc. IEEE International Conference on Computer Communications (INFOCOM), Shanghai, China, Mar. 2012, pp. 268–273

33 Mohamed A., Onireti O., Imran M. A., et al.: 'Control-data separation architecture for cellular radio access networks: a survey and outlook', *IEEE Communications Surveys & Tutorials*, 2016, **18**, (1), pp. 446–465

34 Breslau L., Cao P., Fan L., Shenker S.: 'Web caching and zipfLike distributions: evidence and implications'. Proc. IEEE International Conference on Computer Communications (INFOCOM), New York, USA, Mar. 1999, pp. 126–134

35 Wang X., Li X., Leung V. C. M., Nasiopoulos P.: 'A framework of cooperative cell caching for the future mobile networks'. Hawaii International Conference on System Science (HICSS), Kauai, USA, Jan. 2015, pp. 5404–5413

36 Afanafyev A., Moiseenko I., ZHANG L.: 'ndnSIM: NDN simulator for NS-3, NDN', Technical Report NDN-0005, 2012.

37 Podlipnig S., Boszormenyi L.: 'A Survey of Web Cache Replacement Strategies', *ACM Computing Surveys*, 2003, **35**, (4), pp. 374–398