Cluster-Based Caching Strategy with Limited Storage in Ultra Dense Networks

Chengjia Hu(✉), Xi Li, Hong Ji, and Heli Zhang

Key Laboratory of Universal Wireless Communications, Ministry of Education,
Beijing University of Posts and Telecommunications,
Beijing, People's Republic of China
{hcj, lixi, jihong, zhangheli}@bupt.edu.cn

Abstract. Ultra dense network (UDN) is considered as one of the key techniques to boost the network capacity in 5G. In order to reduce the huge backhaul cost and end-to-end transmission delay, caching the popular content at the edge of UDNs is an inspiring approach. Considering that the storage capacity of a single small base station (SBS) in UDNs is usually limited, SBSs cooperation to store respective file fragments is an interesting approach that needs further investigation. In this paper, we propose a cluster-based caching strategy (CBCS) for limited storage SBSs in UDNs. A novel clustering scheme based on SBSs's load capacity and location is designed with consideration on files fragments and SBSs cooperation. We target the minimum average download delay under the constraint of the number of SBSs in a cluster. The simulation results show that the proposed algorithm could achieve a better hit ratio and has a lower average download delay.

Keywords: Ultra dense network · Clustering · Caching · Download delay · Hit ratio

1 Introduction

With the rapid deployment of intelligent terminals and the exponential growth in network traffic volume, the traditional techniques have proved incapable of satisfying the increasing traffic demand. Ultra dense network (UDN) is considered to be one of the most important techniques to improve capacity and meet the users experience. UDNs generally consists of numerous low power small base stations (SBSs) and deployment density of SBSs is much higher than Long Term Evolution (LTE) networks [1]. However, user will experience unbearable long delay in getting content due to the congestion in backhaul links, it can lead to significant decline in service quality, backhaul capability may become the bottleneck for UDNs. Recent years, caching at the network edge has been paid much attention and become an important way to tackle the backhaul limitation bottleneck

X. Liu et al. (Eds.): ChinaCom 2018, LNICST 262, pp. 116–126, 2019.
https://doi.org/10.1007/978-3-030-06161-6_11

and reduce the latency of services [2]. Nevertheless, considering the deployment costs, the storage capacity of a single SBS in UDNs is very limited. So how to effectively cache files and transmit is an open issue.

To meet the cache limited conditions in UDNs, each SBS stores file fragments instead of full file so each SBS could cache portions of multiple files. Taking into account the densification of UDNs, cluster-based SBSs cooperation cache could further enhance the network performance. There are many existing work related to the clustering and caching issues. In [3], a cluster-based resource allocation strategy is proposed to effectively mitigate the interference and boost energy efficiency (EE) of the UDNs. In [4], in order to maximize the EE of wireless small cell networks, a locally group based on location and traffic load is proposed to couple SBSs into clusters. In [5], the adjacent SBSs are divided into disjoint clusters by using a hexagonal mesh model with distance between cluster centers is 2Rh, and they don't consider the case of empty clusters. In [6], a distributed, cached and segmented video download algorithm based on D2D communication is introduced to effectively improve the throughput in cellular networks. Li et al. [7] proposed a socially aware caching strategy for UDN to improve the network throughput. In [8], a tri-stage fairness algorithm is proposed to solve the resource allocation problem in UDNs. In [9], to resolve the interference and backhaul issues, a backhaul limited cache transmission scheme based on the linear capacity scaling law is proposed in UDN.

So far, there is few well recognized caching strategy for UNDs with limited storage. In this paper, considering the UDNs densification and limited cache capacity, we propose a clustered-based caching strategy (CBCS) to reduce download delay and improve hit ratio. A novel clustering scheme based on SBSs's load capacity and location is introduced with the same SBSs number in each cluster. Sorting the contents according to their popularity, and caching different fragments of the most popular contents in different SBSs whose in the same cluster. By changing the number of SBSs in a cluster, the minimum download delay could be found. Simulation results show that the proposed algorithm could achieve a better hit ratio and has a lower average download delay.

The structure of this paper as follows. The system model and problem formulation is written in Section II. We propose clustering and fragment-based caching strategy in Section III. In Section IV, we describes in detail the simulation results and discussions. Finally, in Section V, we conclude this paper.

2 System Model and Problem Formulation

2.1 Network Model

In Fig. 1, we describe the edge caching network in UDNs in this paper. The network contains S SBSs with U mobile users and one MBS. The users connect to SBSs and SBSs connect to MBS via wireless links. The MBS connection to the core network with wire links. We assume each SBS caches files according to the cache scheme and the size of different files are same.

2.2 SBS Clustering Model

Identifying similarities between SBSs is important step in cluster strategy. The location of SBSs and their load capacity are two important ones of many similarity features. The SBS locations reflect the capability of coordination among nearby SBSs while the SBS load capacity defines mutual interference and the willingness to cooperate [4]. Next, similarities between SBSs based on location and load capacity can be calculated as follows.

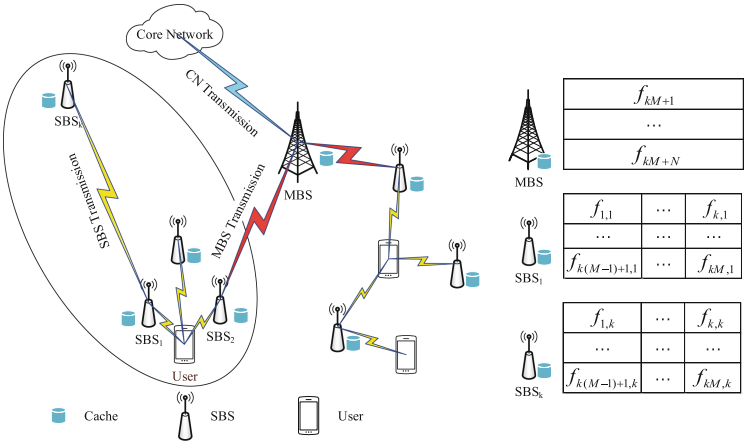

Fig. 1. Edge caching network considered in UDNs

Firstly, based on the distance between SBSs S and S_1, we calculate the Gaussian similarity as follows:

$$X_{SS_1} = \begin{cases} exp(\frac{-||X_S - X_{S_1}||^2}{2\sigma_X^2}) & \text{if } ||X_S - X_{S_1}|| \leq r, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Where σ_X is a constant and X_S is the coordinates of the SBS S in the Euclidean space. The r denote maximum neighborhood distances between SBSs. Furthermore, the value of X_{SS_1} is used as the (S,S_1)-th entry of the distance-based similarity matrix $\mathbb{S}$. As these SBSs get closer, the similarity will increase and SBSs are more likely to cooperate with each other.

Secondly, for the sake of simplified, we assume that the frequency of the user request file is the same. So SBS's load capacity simplified as the maximum number of users which SBS can service. So the load-based dissimilarity between SBSs S and S_1 can be calculated as follows:

$$N_{SS_1} = exp(\frac{-||N_S - N_{S_1}||^2}{2\sigma_N^2}) \tag{2}$$

Where σ_N controls the range of the dissimilarity and N_S is the maximum number of users which SBS S can service. In addition, the load-based dissimilarity matrix $\mathbb{N}$ is formed using N_{SS_1} as the (S,S_1)-th entry. In general, the SBSs with different load capacity yield more benefit by offloading traffic to one another.

Thirdly, the joint similarity matrix $\mathbb{W}$ with W_{SS_1} as the (S,S_1)-th element which combine distance-based similarity with load-based dissimilarity is formulated as follows:

$$W_{SS_1} = (X_{SS_1})^{\theta} \cdot (N_{SS_1})^{(1-\theta)}, \quad \theta \in [0,1] \tag{3}$$

Where θ controls the extent to which the distance and load affect the joint similarity.

Finally, we propose a clustering mechanisms based on a similarity matrix $\mathbb{W}$. A SBS was randomly selected as the center of the initial cluster and it find the most similar k the SBSs form a cluster according to the similarity matrix $\mathbb{W}$. Choose remainder SBS which closest to the previous cluster center as the center of the new cluster. It will select SBS which already within the cluster to ensure every cluster has the same number of SBSs. Once clusters are formed, the SBS which load capacity is best within the cluster is selected as the cluster head. The cluster head is used to allocating resources within a cluster and coordinate the transmissions between the cluster members.

2.3 Caching and Transmission Model

In Fig.1, we also describe the specific caching scheme and transmission model. We consider a finite and sorted by popularity content library $\mathbb{F} = \{f_1, f_2, \cdots, f_i, \cdots, f_I\}$ in UDNs, where f_i is the i-th most popular file and the size of each file are F. Each SBS can store up to M files and MBS can be cached $N(M < N < T)$ files. A cluster can cache kM files with k cooperative SBSs. We choose kM of the most popular file and each file (e.g. video) is divided into k fragments and caching in the different SBS. Moreover, files f_i with popularity order $kM < i \leq kM + N$ are cached in MBS and the remaining files are not cached.

When a user requests file from $\mathbb{F}$, he or she can obtain file mainly through the following three ways.

(1) SBS Transmission: When the file with popularity order $1 \leq i \leq kM$, k SBSs in the cluster has different fragments of this file. On the one hand, fragments are transmitted to the user directly by the SBS (e.g. SBS_1 and SBS_2) which cover the user. On the other hand, cluster head scheduling remaining SBSs (just like SBS_k) by relay sending fragments to the user.

(2) MBS Transmission: When the file with popularity order $kM < i \leq kM + N$ which caching in MBS, the SBS needs to get this file from MBS and send it to the user.

(3) Core Network Transmission: When the file not cached, the SBS needs to get this file from the core network and send it to the user.

2.4 Average Hit Ratio

We assume each local user issues an independent request for a file whose probability is related to Zipf-like popularity distribution [10]. A user has the probability p_i of requesting the i-th file. We can get p_i through

$$p_i = \frac{1/i^{\alpha}}{\sum_{j=1}^{I}(1/j^{\alpha})} \tag{4}$$

Where α is a decay constant. So the SBS expected hit ratio is calculated as

$$P_{hit}^{S} = \sum_{i=1}^{kM} p_i \tag{5}$$

It is obvious that the cache hit ratio will increase if the number of SBSs in a cluster increase.

2.5 Average User Download Delay

Let σ^2 be the noise power. The $h_{s,u}$, $h_{M,s}$ means the channel gain between SBS s to user u and the MBS to SBS s, respectively. Denote the transmission power of the SBSs, the MBS as P_S and P_M respectively.

In order to simplify the calculation, we assume that MBS and SBSs not reuse spectrum resources and the MBS has the bandwidth ω_M. The SBSs which in a cluster allocate orthogonal frequency band ω_S. So the signal-to-noise ratio $SNR_{s,u}$ between SBS s and user u and the $SNR_{M,s}$ between MBS and SBS s could be denoted as $SNR_{s,u} = \frac{P_S \cdot h_{s,u}}{\sigma^2}$ and $SNR_{M,s} = \frac{P_M \cdot h_{M,s}}{\sigma^2}$. Thus we would get the user u download $Delay_1^u$ of SBS transmission and the $Delay_2^u$ of MBS transmission. The expression of $Delay_1^u$ is

$$Delay_1^u = \arg\max_{s=1,2,\cdots,t} \frac{N_s F/k}{\frac{\omega_S}{k} \cdot \log_2(1 + SNR_{s,u})} \tag{6}$$

Where N_s is the number of fragments which SBS s required to transmit. Denote the number of SBSs which directly linked to the user as t.

The $Delay_2^u$ contains the delay of SBS receiving file from MBS and SBS sending the file to the user. We can get $Delay_2^u$ through

$$\begin{aligned} Delay_2^u = & \frac{F}{\omega_M \cdot \log_2(1 + SNR_{M,s})} \\ & + \frac{F}{\omega_S \cdot \log_2(1 + SNR_{s,u})} \end{aligned} \tag{7}$$

When a user requests for the files which are not cached. The user download delay contains $Delay_2^u$ and the $delay_{c,M}$ of core network sending file to MBS. To

simplify, we denoted $delay_{c,M}$ as C which is a constant. Therefore, the $Delay_3^u$ is calculated as

$$Delay_3^u = Delay_2^u + C \tag{8}$$

The user obtain files probability P_1^u through SBS transmission, P_2^u through MBS transmission and P_3^u through core network transmission could be denoted as $P_1^u = P_{hit}^S$, $P_2^u = \sum_{i=kM+1}^{kM+N} p_i$ and $P_3^u = 1 - P_1^u - P_2^u$ respectively. The theoretical expression of user u average download $Delay^u$ is

$$Delay^u = P_1^u \cdot Delay_1^u + P_2^u \cdot Delay_2^u + P_3^u \cdot Delay_3^u \tag{9}$$

3 The Proposed CBCS Algorithm

In this section we rearrange the CBCS algorithm ideas. For SBSs, we propose a clustering mechanisms based on SBSs's location and load capacity in which every cluster has the same number of SBSs. For users, we count and sort the popular files that they might need. Furthermore, we can calculate each requiring probability of the files by (4). We would give the maximum number K of SBSs in a cluster so as to find the minimum value of average download delay conveniently.

After all the SBSs have been successfully clustered, and popular files have been cached in the appropriate location according to the cache strategy, then the user starts to access the SBS and requests files. Firstly, the connected user check whether required files already cached in SBSs. User can obtain required files according to SBSs transmission if required files already cached in SBSs. It follows MBS transmission if this file cached in MBS. Otherwise, it follows core network transmission. Then, the average download delay can be calculated according to (9). Our goal is to minimize the average download delay under the constraint of the number of SBSs in a cluster. The optimization problem is accordingly formulated as:

$$\begin{aligned} \underset{k}{\text{minimize}} \quad & Delay^u \\ \text{subject to:} \quad & k \in \{1, 2, \cdots, K\} \end{aligned} \tag{10}$$

Where K is the maximum number of SBSs in a cluster. Change the number of SBSs in a cluster, the minimum average download delay could be found. In Algorithm 1, we summarize the process of the CBCS algorithm as follows.

4 Simulation Results And Discussions

In this section, we evaluate the performance of the CBCS algorithm and analyze the impact of important parameters through simulation. For comparison, we simulated the "most popular" placement (MPP) scheme which each SBS cache the M most popular files mentioned in [11].

In the simulation, we let $F = 10M$ bits, $\alpha = 0.8$, $I = 200$, $M = 10$, $N = 100$, and $\beta = 4$ if there are no special instructions. Simulation area size of 200 m*200 m and we put the MBS in the center. 50 SBSs are random distribution and coverage area radius r is 50 m. The values of P_S and P_M are 100 mW and 20W respectively, and $L_0 = -30$ dB. The value of σ^2 is -100 dBm and bandwidth $\omega_M = \omega_S = 1$ MHz [12]. We assume the value of $delay_{c,M}$ as C is 1 s.

We can get the relationship between average hit ratio and the number of SBSs in a cluster from Fig. 2. As can be seen, increasing the value of k is beneficial to increase the cache hit ratio, and the growth rate becomes smaller. The larger the value of k is, the more files are stored in a cluster and therefore the SBS hit ratio would increase. However, file transmission collaboration between SBSs becomes

Algorithm 1 Minimizing Average Download Delay in UDNs via CBCS

Initialization:
(a) Set S SBSs coordinates X_S and load capacity N_S subject to random distribution. Set constant σ_X,σ_N,r,θ;
(b) SBS storage capacity M, MBS storage capacity N, number of all files I, size of the files F;
(c) SBS transmission power P_S, transmission power P_M, noise power σ^2, channel gain relevant parameters L_0 and β, SBSs cluster bandwidth ω_S and MBS bandwidth ω_M, set K.
Calculate matrix $\mathbb{W}$ according to (1),(2),(3) and select initial SBS.
for $k = 2, \cdots, K$ **do**
    while the number of remainder SBSs $\neq 0$ **do**
        Find the number n of remainder SBSs which the similarity with cluster center is greater than 0.
        if $n > k - 1$ **then**
            Find most similar k SBSs form a cluster.
        else
            Select SBS already within the cluster to ensure every cluster have the k number of SBSs.
    The file placement strategy shown in Fig.1
    The users connect to the SBSs.
    for $u = 1, \cdots, U$ **do**
        User U_u begin to require files.
        if this file is stored in SBSs **then**
            Calculate the download $Delay_1^u$.
        else
            if this file is stored in MBS **then**
                Calculate the download $Delay_2^u$.
            else
                Calculate the download $Delay_3^u$.
        Calculate the user u average download $Delay^u$.
    Calculate the average download $Delay$.
    Calculate the SBS expected hit ratio P_{hit}^S.
Output the $Delay_{min}$ and P_{hit}^S.

complicated. Thus, the higher average download delay appears. Further, the smaller the value of I is, the higher the hit ratio.

Figure 3 indicates the changes of average download delay with different number of SBSs in a cluster. In Fig. 3, from 1 to 6 the number of SBSs in a cluster, the user download delay firstly decreased and then increased. The reason for firstly increased is the probability of SBSs transmission increasing whose delay smaller than MBS transmission. Along with the increase of k, SBS cooperative transmission is more and more complex, finally the transmission delay over MBS transmission. Therefore, the user download delay would increase. Furthermore, when $S = 50$, combined with Fig. 2 compare MPP and CBCS ($k = 4$), we can see that average hit ratio increases from 0.35 to 0.6 while average download delay no change. The simulation results verify that our algorithm can bring better cache hit ratio in UDNs.

In Fig. 4, we compare the average download delay performance of CBCS and MPP for different total SBSs. Figure 4 clearly shows that the average download delay decreases when the value of S increases with CBCS. However, the delay basically no change with MPP. This is because we place the user in a fixed position nearby the cluster head to simplify, so the distance between user and SBS is fixed with MPP. The average download delay is not changed. The distance between the user and SBS decreases when the value of S increases with CBCS, as a result, the user download delay is reduced.

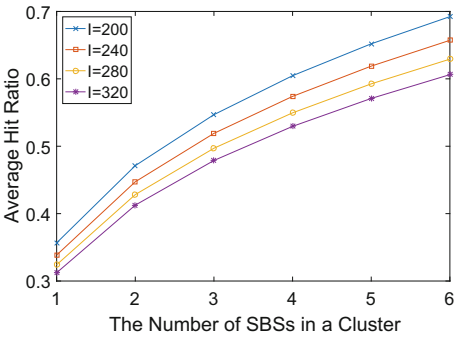

Fig. 2. The average hit ratio with different number of SBSs in a cluster

Figure 5 illustrates the the effect of the Zipf parameter α on the average download delay. We can see from Fig. 5 that when the Zipf exponent increases, the user average download delay decreases. The reason is higher α means that the popularity of the top files in the library will increase, and they are more often required and SBS transmission delay smaller than MBS transmission in general. Furthermore, on account of the SBSs which in a cluster could be allocated orthogonal frequency band ω_S and MPP means that each the bandwidth

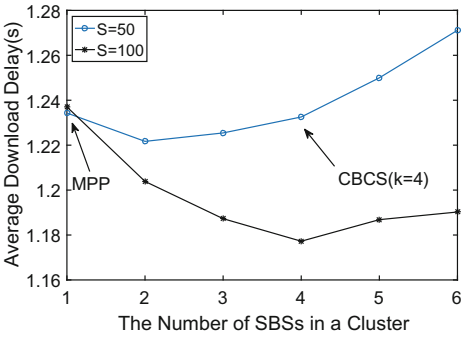

Fig. 3. The average download delay with different k in a cluster

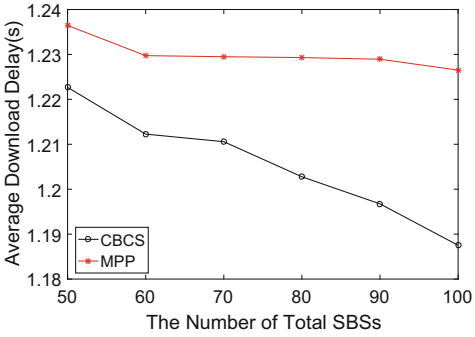

Fig. 4. The average download delay with different total number of SBSs

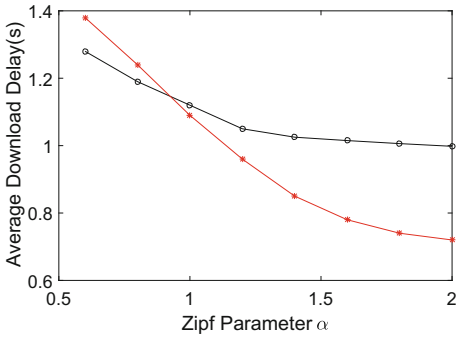

Fig. 5. The average download delay with different Zipf parameter α

of the SBS is ω_S, so MPP less than CBCS SBS transmission delay according to (6). That is the reason for MPP less than CBCS average download delay when $\alpha > 0.9$.

5 Conclusion

In this paper, we focus on the issue of how to cache under limited storage capacity in UDNs. We designed a CBCS algorithm based on SBSs clustering and files fragments. A clustering scheme is proposed based on SBSs's location and load capacity in which every cluster has the same number of SBSs. Then, we split files into fragments and caching them in different SBSs. To find the optimal solution, a traversal algorithm is used to resolve the optimization model. Compared with the MPP algorithm, the simulation results show that the proposed CBCS algorithm can achieve a higher hit ratio under similar average download delays. For future work, in order to improve the performance of the system, variable number of SBSs inside the cluster would be considered, as well as the user movement in UDNs.

Acknowledgements. This paper is sponsored by National Natural Science Foundation of China (Grant 61771070 and 61671088).

References

1. Kamel, M., Hamouda, W., Youssef, A.: Ultra-dense networks: a survey. IEEE Commun. Surv. Tutor. **18**(4), 2522–2545 (2016). Fourthquarter
2. Zhang, J., Zhang, X., Wang, W.: Cache-enabled software defined heterogeneous networks for green and flexible 5g networks. IEEE Access **4**, 3591–3604 (2016)
3. Samarakoon, S., Bennis, M., Saad, W., Latva-aho, M.: Dynamic clustering and on/off strategies for wireless small cell networks. IEEE Trans. Wirel. Commun. **15**(3), 2164–2178 (2016)
4. Chen, Z., Lee, J., Quek, T.Q.S., Kountouris, M.: Cluster-centric cache utilization design in cooperative small cell networks. In: 2016 IEEE International Conference on Communications (ICC), pp. 1–6 (2016)
5. Huang, K., Andrews, J.G.: An analytical framework for multicell cooperation via stochastic geometry and large deviations. IEEE Trans. Inf. Theory **59**(4), 2501–2516 (2013)
6. Al-Habashna, A., Wainer, G., Boudreau, G., Casselman, R.: Distributed cached and segmented video download for video transmission in cellular networks. In: 2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), pp. 1–8 (2016)
7. Zhu, K., Zhi, W., Chen, X., Zhang, L.: Socially motivated data caching in ultra-dense small cell networks. IEEE Netw. **31**(4), 42–48 (2017)
8. Hao, P., Yan, X., Li, J., Li, Y.N.R., Wu, H.: Flexible resource allocation in 5g ultra dense network with self-backhaul. In: 2015 IEEE Globecom Workshops (GC Wkshps), pp. 1–6 (2015)
9. Liu, A., Lau, V.K.N.: How much cache is needed to achieve linear capacity scaling in backhaul-limited dense wireless networks? IEEE/ACM Trans. Netw. **25**(1), 179–188 (2017)
10. Breslau, L., Cao, P., Fan, L., Phillips, G., Shenker, S.: Web caching and zipf-like distributions: evidence and implications. In: INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings, vol. 1, pp. 126–134. IEEE (1999)

11. Gabry, F., Bioglio, V., Land, I.: On energy-efficient edge caching in heterogeneous networks. IEEE J. Sel. Areas Commun. **34**(12), 3288–3298 (2016)
12. Jia, C., Lim, T.J.: Resource partitioning and user association with sleep-mode base stations in heterogeneous cellular networks. IEEE Trans. Wirel. Commun. **14**(7), 3780–3793 (2015)