Distributed uplink cache for improved energy and spectral efficiency in B5G small cell network

Mubarak Mohammed Al Ezzi Sufyan¹, Waheed Ur Rehman¹, Tabinda Salam², Abdul Rahman Al-Salehi³, Qazi Ejaz Ali¹, Abdul Haseeb Malik¹

¹ Department of Computer Science, University of Peshawar, Peshawar, Pakistan, ² Department of Computer Science, Shaheed Benazir Bhutto Women University Peshawar, Peshawar, Pakistan, ³ Department of Electronics Engineering, International Islamic University, Islamabad, Pakistan

* wahrehman@uop.edu.pk

Abstract

The advent of content-centric networks and Small Cell Networks (SCN) has resulted in the exponential growth of data for both uplink and downlink transmission. Data caching is considered one of the popular solutions to cater to the resultant challenges of network congestion and bottleneck of backhaul links in B5G networks. Caching for uplink transmission in distributed B5G scenarios has several challenges such as duplicate matching of contents, mobile station’s unawareness about the cached contents, and the storage of large content size. This paper proposes a cache framework for uplink transmission in distributed B5G SCNs. Our proposed framework generates comprehensive lists of cache contents from all the Small Base Stations (SBSs) in the network to remove similar contents and assist uplink transmission. In addition, our framework also proposes content matching at a Mobile Station (MS) in contrast to an SBS, which effectively improves the energy and spectrum efficiency. Furthermore, large size contents are segmented and their fractions are stored in the distributed cache to improve the cache hit ratio. Our analysis shows that the proposed framework outperforms the existing schemes by improving the energy and spectrum efficiency of both access and core networks. Compared to the existing state of the art, our proposed framework improves the energy and spectrum efficiency of the access network by 41.28% and 15.58%, respectively. Furthermore, the cache hit ratio and throughput are improved by 9% and 40.00%, respectively.

1 Introduction

In recent years, content-based services such as video streaming are exponentially growing. In addition, the number of internet users by 2023 is expected to be 5.3 billion [1]. This will tremendously increase the network volume for both uplink and downlink in Beyond 5th Generation (B5G) networks. It will also result in several challenges such as traffic load, congestion, and latency, in addition to, significant consumption of energy and spectrum [2–4]. To solve
the challenges incurred by this data explosion, the SCN is widely advocated in B5G networks. However, it affects the users’ data requirements over time and location. In addition to, creating bottlenecks at the backhaul. It is caused as a result of a significantly improved data transmission rate in the SCN. Another way to cope with the data explosion is the use of caching in the network. In this technique, cached versions of the contents are stored in the network at different locations such as Base Station (BS), Macro Base Station (MBS), SCN, and/or Mobile Stations (MSs). If the cached versions are available, devices do not need to upload or download the contents to/from the core network. In addition, as the contents are available locally, it reduces the cost of communication, energy/bandwidth consumption and latency [5–8].

The rapid growth of the SCNs also increases the complexity of the cellular network, especially in a cooperative scenario. In such a case, the distributed cache can be employed to act as a single cache but span across the network. A distributed cache can be seen as an extension of a single cache. It provides a higher data rate with less energy and bandwidth consumption along with low latency to access contents in order to meet users’ quality of experience [9–11]. Distributed caching may be employed for both downlink and uplink transmission depending upon the demands of the MSs [12]. Downlink caching restrains the unnecessary download of content by providing it locally based on its popularity to reduce the congestion on the backhaul link and core network. Whereas, uplink caching is used to store the MS’s data intended to be uploaded with the constraints of avoiding unnecessary upload to reduce the data traffic in the network, especially the access network link [13, 14]. Distributed caching can be deployed in different types of network such as content-centric network [15], cloud-small cell networks [16, 17], wireless cellular networks [18], dense-SBSs [19], an edge network [20], B5G mobile edge computing [6], HetNets SBs [21], B5G relaying networks [22], Small Cell Base Station (SCBS) [23], multi-antenna SCNs [24], and 5G-SBS [25].

Recently, a lot of research work is done on addressing the challenges of distributed caching in a cellular network with some work on uplink caching [26–33]. The authors in [26], proposed a novel upload cache architecture to support the parallel uploading of segmented files. The authors in [27], studied the energy-efficient cooperative coded caching issue in heterogeneous SCNs to minimize the energy consumption for content caching and content delivery. The authors in [28], proposed an uplink cache system in delay tolerant SCNs. The main focus in [28] is to analyze the cache size and its effectiveness for uplink caching. Furthermore, duplicate elimination of contents is performed at SBS via matching the hash key of chunks of the file after uploading the real content, which is not very practical for an uplink scenario. The authors in [29] developed a novel multiple-input multiple-output (MIMO) network architecture with a large number of BSs employing cache-enabled uplink transmission. The authors proposed the modified von Mises distribution as a popularity distribution function and derive the outage probability, in addition, identified the relationship between cache storage and outage probability to be directly proportional. Therefore, it is found that the delivery rate also improves with the increase of cache storage space in addition to a denser network. The authors in [30] have proposed an innovative approach to reduce peak traffic by utilizing distributed cache of Internet of Things (IoT) devices. The proposed scheme employed an uplink transmission scheduling based on delay adaptation to mitigate the sporadic access network congestion. The average delivery latency is investigated by the authors in [31], where they proposed a proactive caching technique for movie on-demand streaming on the internet. The probabilistic caching is studied in [32], where authors proposed an optimized caching strategy to improve the successful download probability. Furthermore, in [33], the authors presented the contents matching the mobile data to be uploaded at an MS level before the real transmission of the actual content taken place to uploading the dissimilar contents only without considering the cooperative scenario.
All the aforementioned research papers proposed novel uplink cache architectures and schemes to effectively improve the network performance. However, these papers have not presented the cache benefits on Energy Efficiency (EE) and Spectral Efficiency (SE). In addition, the existing literature proposes to use an SBS for performing content matching [28, 29], which entails that the MSs will be unaware of cached contents. This will lead to unnecessary upload of content, which is not desirable. Furthermore, the existing literature has also identified that the large content size significantly degrades the performance of the distributed cache. In addition, the existing works in [28, 29] have not considered content duplication and its effect on energy and spectral efficiency in a distributed scenario. The authors in [33] considered content matching at an MS mitigating the duplicate upload, however, it did not consider a distributed scenario. As well as, the content segmentation and its effective placement in the context of distributed cache is not considered in the existing literature. This motivates us to propose a novel cache-enabled uplink transmission framework that addresses all these challenges. The main objective of this work is to improve energy and spectral efficiency along with improving cache performance in a distributed scenario. Firstly, our proposed framework generates an unduplicated list of cache contents in the distributed scenario in an attempt to be used as a map for an MS to decide whether to upload the contents or not. Secondly, it proposes the scheme to perform content matching at an MS and lastly, it performs the content segmentation to divide a large size content into smaller for effective storage across the distributed network. The main contributions of this paper can be summarized as follows:

1. We proposed a novel scheme to effectively generate a disparate list of distributed cache for uplink transmission. The proposed scheme leverages cooperative communication to efficiently generate a list of contents based on popularity and content validity. This way the list is updated with the most relevant content, which improves the cache hit ratio.

2. An efficient content matching scheme is also proposed to facilitate an MS to corroborate the cached contents. This will significantly limit the amount of uplink transmission, which will improve the EE and SE.

3. We also proposed a content segmentation with distributed placement scheme, which improves the content placements in the distributed cache network by splitting the larger contents into smaller. The proposed scheme significantly improves the cache hit ratio, which intrinsically improves the EE and SE.

4. We compare our proposed framework with state-of-the-art architectures and algorithms to analyze its effectiveness. We mainly focused on energy consumption and spectrum efficiency, which is the main limiting factor of the existing literature.

This paper is organized as follows: Related work is discussed in Section 3. The system model is discussed in Section 4. Section 5 presented our proposed framework. Experimental, Performance evaluation and conclusion are discussed in Sections 6 and 7, respectively.

2 Related work

The use of cache for the downlink transmission is widely researched. Its effectiveness is broadly accepted, especially the energy and spectral efficiencies, which is summarized in Table 1, with some work related to distributed cache in Table 2. However, cache-enabled uplink is researched to a lesser extent. The state-of-the-art and our reference points are in [28, 29]. In [28], the authors presented a framework to relieve the burden of wireless SCNs by considering cache-enabled uplink transmission in a delay-tolerant network. The paper also proposed duplicate content matching at an SBS by comparing the hash key of chunks of the file...
| Ref. | Year | Caching at | Contributions | Components | Modeling Tools | Performance Measurements |
|-----|------|------------|---------------|------------|---------------|-------------------------|
| [27] | 2017 | SBS, MBS  | Proposed a cooperative coded caching scheme to obtain EE by reducing the Energy Consumption (EC) by content caching and delivery, and transport of cooperative, and backhaul, in addition, proposed a greedy-based caching placement algorithm to optimize the caching placement that improved the content delivery efficiency and enhanced the Quality of experience (QoE) for end-users. | Het.SCN consists of SBS, MBS, Users | Maximum-Distance Separable (MDS), Zipf | EC |
| [34] | 2017 | SCBS      | Investigated the system throughput by analyzing the EE of cache-enabled cooperative Dense-SCNs (D-SCNs) using an affinity propagation algorithm to divide the SCBSs into different clusters and derivation the closed-form of the EE of Coordinated Multi-Point-Joint Transmission (CoMP-JT) cache-enable cooperative D-SCNs to reduce the EC of transmitting, circuit powers, and caching. | D-SCN consists of SCBS, MBS, User | PPP | EE |
| [35] | 2018 | MBS, SBS  | Proposed a Scalable Video Coding (SVC) scheme based fractional caching for providing on-demand High Definition Video (HDV) with different perceptual qualities to solve the issue of repeated content deliveries and then derivation the expressions for successful transmission probabilities and ergodic service rates based on stochastic geometry to establish the integrated power consumption model and using the standard gradient projection method to optimize EE. | HetNets consist of MBS, SBS, Users | Stochastic Geometry, hPPP, Zipf | Successful transmission probability, EE |
| [36] | 2019 | MBS, SBS  | Proposed an energy-efficient collaborative caching scheme to minimize the energy consumption by considering the constraints of limited storage capacity, content popularity, placement, and access to contents to provide efficient utilization of hotspot cache, and using knapsack with Zipf to ensure the local availability of cache contents and minimizing transportation cost. In addition, the proposed scheme is to support the Device-to-Device (D2D) communication. | 5G-WN consist of MBS, SBS, Users | Zipf and Knapsack | EC of backhaul, hotspot, and the overall network, Transport energy cost |
| [37] | 2019 | MBS, SBS  | Investigated a file placement strategy in cooperative SCN to maximized EE via proposed an energy-efficient cooperative caching scheme (EECCS) with the capability of accessing the SBS cache contents by other SBSs to ensure the caching service for all the users. | SCN consists of MBS, SBS, users | Additive White Gaussian Noise (AWGN) | EE |
| [38] | 2020 | MBS, SBS  | Propose a sleeping mechanism in the 5G-SCNs with Energy Harvesting (EH) function with jointly cooperative caching to minimize EC and backhaul traffic by the binary particle swarm optimization (BPSO) to turn the SBS to sleep mode at every time slot based on the battery capacity and fewer contents delivery, in addition, an Iterative Algorithm With Status Update and Cooperative Caching to optimize the content caching placement. | 5G-SCN consist of MBS, SBS, Users | Zipf, MDS | EC of SBSs, backhaul and power grid |
| [39] | 2020 | MBS, mmSBS, and UE | Analyzed the successful probability for users to obtain files by different communication modes and consider the energy costs and offloading probability as the objective function to optimize the joint caching policy (Random caching strategy (RCS), and File popularity-based random strategy (FPRS)) of the user equipment (UE) and mmWave SBSs (mmSBSs) to improve the file transmission efficiency. | MBS, mmSBS, UE | RCS, FPRS, Zipf, hPPP | Energy cost and Offloading Probability |

**Spectral Efficiency**

(Continued)
after uploading real content. The authors also used First-Input-First-Output (FIFO), random and probabilistic content scheduling strategies for cache management. It enabled an SBS to eliminate the redundancy among users’ uploaded contents to improve the network transmission efficiency. Similarly, in [29], the authors presented a cache-uplink framework for HetNet
with stochastic distributed BSs for temporary caching of user-generated contents. The authors also presented the relationship between cache storage space and outage probability. While the authors in [33], presented a Broadcast cache assist uplink (BCAU) scheme to perform a matching among the attributes of new content and cache contents at an MS to discard the uploading of the available content in the SBS cache before the real transmission of the actual content taken place to improve the EE and SE of uplink transmission over B5G-SCN but the authors is not considering the cooperative among the distributed cache. However, these papers have not considered EE and SE of distributed cache. In addition, content matching at an SBS, as proposed in [28, 29, 33], entails unnecessary uploads from the MSs. Lastly, the challenges of large size contents are not addressed in these papers.

3 System model
This section gives a detailed description of the system model. The cooperative distributed network with cache-enabled SBSs and MBS along with MSs is considered as shown in Fig 1.

3.1 Network model
We consider a cellular network that consists of a cloud, a TDD MBS, \( M \) TDD SBSs, and \( N \) MSs as shown in Fig 1. An MBS is a BS of a cellular network, which is donated by \( G \). An MBS is employed to collect the relevant information from all the SBSs in addition to controlling them. The SBSs and MSs are spatially distributed according to two independent homogeneous Poisson Point Process (hPPP) \( \Phi_B \) and \( \Phi_U \) with the density of SBSs and MSs are represented as \( \lambda_B \) and \( \lambda_U \), respectively. The set of SBSs is indicated by \( B = \{ B_j : j = 1, 2, \ldots, M \} \) serving a set of MSs \( U = \{ U_i : i = 1, 2, \ldots, N \} \). All SBSs are connected to the MBS and a cloud through a backhaul link. While each SBS is bidirectionally connected with its neighboring SBSs via X2 interface (a sidehaul wireless link). All SBSs are in an active mode to be associated with at least one MS to serve. Each MS selects its local SBS based on its propagation distance. Therefore, the set of MSs, which serves by an SBS \( B_j \) is denoted by \( U_{B_j} : 1 \leq i \leq n, 1 < j < M, n < N \), where

| Table 2. Summary of cooperative distributed cache methods. |
|---|
| Ref. | Cache placed | Proposed Scheme |
|---|---|---|
| [16] | SCN | Maximum Distance Separable (MDS) Code. |
| [47] | 5G-SCBS | Decentralized Collaborative Video Caching. |
| [48] | SBS | Encoded into multiple resolutions with multiple Bitrate videos. |
| [49] | Wireless net | Rolling-horizon collaborative cache optimization scheme. |
| [50] | MSNs | Social-aware cooperative caching mechanism. |
| [51] | MNC | A reinforcement learning (RL)-based online learning algorithm to search the optimal caching policy. |
| [52] | MEC | A Group Behavior and Popularity Prediction based Collaborative Caching (GPCC) strategy based on UDR (User Detail Record). |
| [48] | SBS | Cooperative caching at the edge in SBS based on a Bayes-based learning algorithm. |
| [53] | ICN | Edge-oriented Collaborative Caching (ECC). |
| [54] | MEN | Hybrid collaborative caching (hy-CoCa). |
| [55] | SCBS | A rolling horizon cache optimization scheme. |
| [56] | SBSs | Collaborative content caching among multiple SBSs. |
| [57] | HetsNet | Spatially cooperative caching strategy for a two-tier HetsNet consisting of edge servers and caching helpers. |
| [58] | SBS | A backhaul-based cooperative caching scheme that groups several SBSs. |

https://doi.org/10.1371/journal.pone.0268294.t002
i represents an MS served by an SBS $B_j$. In this work, we consider the uplink transmission scenario.

### 3.2 Cache model

We consider $M + 1$ caches across the cellular network. The rest of the characteristics are as under:

#### 3.2.1 Cache storage. The set of SBSs caches is denoted by $C = \{C_{B_j} : j = 1, 2, \ldots, M\}$. Each cache stores a set of popular contents denoted by $F = \{F_l : l = 1, 2, \ldots, w\}$, where $w$ is the total number of cached contents. The SBS’s cache $C_{B_j}$ stores a set of popular contents denoted by $C_{B_j,l} = \{C_{B_j,l,1}, C_{B_j,l,2}, \ldots, C_{B_j,l} : 1 \leq j \leq M, 1 \leq l \leq w\}$, where $l$ is the total number of the cached contents of $B_j$. The popularity of content $(C_{B_j,l})$ is denoted by $q^{C_{B_j,l}}$ and is modeled by Zipf distribution according to [38]

$$q^{C_{B_j,l}} = \frac{r_{C_{B_j,l}}^{-\delta}}{\sum_{l=1}^{w} C_{B_j,l}^{-\delta}}$$

where $r_{C_{B_j,l}}$ represents the popularity rank of the content $C_{B_j,l}$, and $\delta$ is the skewness of the popularity distribution.

Each cached content has some attributes such as name, size, hash key, length, etc. The set of attributes of each cached content is denoted by $P(C_{B_j,l,\kappa}) = \{P(C_{B_j,l,\kappa,1}), P(C_{B_j,l,\kappa,2}), \ldots, P(C_{B_j,l,\kappa,\kappa})\}$, where $\kappa$ is the serial number of content in a cache and $\kappa$ is its maximum number of attributes. The attributes of each cached content will be used for matching to determine the duplicate content and eliminate it (More details in section 4).
The total storage capacity $W$ of all $M + 1$ cache in the network can be represented as:

$$W = S(G_C) + \sum_{j=1}^{M} S(C_{B_j}), 1 < j \leq M,$$

(2)

where $S(G_C)$ is the storage capacity of MBS’s cache and $S(C_{B_j})$ is the storage capacity of an SBS’s cache.

3.2.2 Distributed caching model. We are considering a cooperative distributed cache, where caches are located at the SBSs and MBS, which appear to be working as a single cache. The contents are stored in a distributed manner across the network. The rationale of this content segmentation is based on [35, 43], where authors argued that the larger size of the contents significantly affects the cache effectiveness due to the scarcity of the storage space.

Therefore, we have proposed that small-sized contents may be stored locally at an SBS, while large size contents can be split into $Q$ segments [27, 38, 56].

The set of segments of a content can represented as $\tau_{C_{B_j},l} = \{(C_{B_j,l}), (C_{B_j,l})_2, \ldots, (C_{B_j,l})_Q\}$ for all $1 \leq l \leq w$, where the size of each segment is determined based on the content’s size ($S(C_{B_j,l})$) and the free space of the local SBS and its neighboring. The segment(s) stored in a separate cache create(s) $E_Q$ encoded packets for each segments as $E_Q = \{e_1^1, e_1^2, \ldots, e_Q^1\}$ using MDS code [59], where $e_1^1$ represents encoded packets for $(C_{B_j,l})_1$ and so on. Furthermore, if the content is too large, it will be cached at the MBS’s cache as a whole content. The placement of segment(s) is determined in each cache by using the Hash key $H(C_{B_j,l})$ of the whole content.

3.2.3 Cache availability and efficiency. Cache availability and efficiency can be evaluated using the cache hit ratio and cache miss ratio. A cache hit occurs, when the content is available in the cache, otherwise, it is a cache miss. According to [60, 61], a cache hit and miss of each SBS $B_j$ are given as:

$$\text{Hit}_{C_{B_j}}^n = \left( \frac{\text{Hit}_{C_{B_j}}}{\text{Hit}_{C_{B_j}} + \text{Miss}_{C_{B_j}}} \right),$$

(3)

where $\text{Hit}_{C_{B_j}}$ and $\text{Miss}_{C_{B_j}}$ are the counter of the cache hit and cache miss of SBS $B_j$, respectively.

$$\text{Miss}_{C_{B_j}} = (1 - \text{Hit}_{C_{B_j}}),$$

(4)

In the scenario of the cache-enabled uplink, the availability of data in a cache is categorized as follows:

1. $\text{Hit}_{C_{B_j}} = 1$: New content is available in distributed cache, therefore, an MS send Message of Target Destination (MoTD)(More details on MoTD in Section(4.2)) rather than actual content.

2. $\text{Hit}_{C_{B_j}} = 0$: New content is unavailable in the distributed cache, therefore, new content will be uploaded.
3.3 Communication model

The uplink capacity of an MS $U_{ij}$ denoted by $\mathcal{N}_{ij}^{UL}$, and can be represented as \cite{62, 63}

$$\mathcal{N}_{ij}^{UL} = B \log_2 \left(1 + \text{SNR}_{ij} \right),$$

(5)

where $B$ represents the channel bandwidth and SNR$_{ij}$ is the signal-interference-to-noise ratio (SINR) of the received signal from MS $U_{ij}$ at its serving SBS $B_j$ and can be represented as,

$$\text{SNR}_{ij} = \frac{TP_{ij}^{ul} H_{ij}^{ul} || d(U_{ij}, B_j) ||^{-\alpha}}{\sum_{m \neq j} I_i + \sigma^2},$$

(6)

where $TP_{ij}^{ul}$ is the uplink transmit power of MS $U_{ij}$, $H_{ij}^{ul}$ is the corresponding uplink channel gain. $|| .||$ stands for Euclidean norm and $d(U_{ij}, B_j)$ is the separation distance between $U_{ij}$ and $B_j$. $\alpha$ is the path loss exponent. $I_i$ is a set of interfering MSs. $\sigma^2$ is the noise power spectral density at an MS.

Using (5) and (6), the (5) can be rewritten as;

$$\mathcal{N}_{ij}^{UL} = B \log_2 \left(1 + \frac{TP_{ij}^{ul} H_{ij}^{ul} || d(U_{ij}, B_j) ||^{-\alpha}}{\sum_{m \neq j} I_i + \sigma^2} \right),$$

(7)

Therefore, the uplink network capacity is the sum of cache hit and miss ratios of both the access link (MS) and backhaul (SBS) and is given by:

$$\mathcal{N}_{Net}^{UL} = \left\{ \begin{array}{c}
\mathcal{N}_{MS}^{UL} \hfill \\
\mathcal{N}_{SBS}^{UL} \hfill 
\end{array} \right\},$$

(8)

where $TP_{B_j}^{ul}$ is the uplink transmission power of the SBS. $H_{B_j}^{ul}$ is the corresponding UL channel gain by an MBS/C-RAN. $|| .||$ stands for Euclidean norm and $d(B_j, G)$ is the separation distance between $B_j$ and $G$.

3.4 Energy consumption model

In this subsection, we describe the Energy Consumption (EC) of an MS, an SBS, and a cache as follows,

3.4.1 Mobile station energy consumption ($EC_{ij}$). The MS $EC_{ij}$ is given according to \cite{33, 64}:

$$EC_{ij} = EC_m + EC_{op} + ET_{ij}^{ul} + EC_r,$$

(9)

where $EC_m$ and $EC_{op}$ are energy consumption of performing matching and other operations, respectively. $EC_r$ is energy-consuming for receiving the packet(s) of an SBS’s reply. $ET_{ij}^{ul}$ is the energy consumption for transmitting data by the MS. In case of cache hit, $ET_{ij}^{ul}$ is only
calculated for the communication cost of sending the MoTD message to an SBS, else $ET_{U_i, j}$ is calculated for the communication cost of sending the whole content.

According to (9), the average EC of the MSs ($EC_{avg}$) is given as,

$$EC_{avg} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} EC_{U_i, j}}{N} \quad (10)$$

### 3.4.2 Caching energy consumption ($C_{EC}$)

The caching energy consumption $C_{EC}$ is the energy spent by the cache to perform the different operations such as cache hit/miss, cache management, cooperation among distributed cache and is given according to [65] as:

$$C_{EC}^{B_j} = \left\{ \begin{array}{l} Hit_{B_j}^{C_{hit}} \times EC_{hit} + Miss_{B_j}^{C_{miss}} \times EC_{Miss} \\ EC_{Hit} = EC_{bus} + EC_{cell} \\ EC_{Miss} = EC_{bus} + EC_{cell} + EC_{pad} + EC_{chip} \end{array} \right\}$$

where $EC_{hit}$ and $EC_{Miss}$ are the energy consumption of cache hit and miss, respectively. $EC_{bus}$, $EC_{cell}$, $EC_{pad}$, and $EC_{chip}$ are energy consumed of address and data bus, sum cell arrays, address and data pads of processor and off-chip cache, respectively. More detail about the calculation of the $EC_{bus}$, $EC_{cell}$, $EC_{pad}$, and $EC_{chip}$ in [65].

In the case of cooperative distributed caching, an SBS’s cache not only consumes energy for data exchange but also for content partitioning. In addition, according to the proposed framework in Section(4), each SBS generates and broadcasts the list of its cache contents to facilitate content matching. Energy consumption of all these operations is considered to compute $EC_{CDC}$.

Therefore, (11) can be rewritten as,

$$C_{EC}^{B_j} = \left\{ \begin{array}{l} Hit_{B_j}^{C_{hit}} \times EC_{Hit} + Miss_{B_j}^{C_{miss}} \times EC_{Miss} + EC_{CDC} \\ EC_{Hit} = EC_{bus} + EC_{cell} + EC_{pad} + EC_{chip} \end{array} \right\}$$

where the $Miss_{B_j}$ and $Hit_{B_j}$ of an SBS $B_j$ are given according to (3) and (4).

### 3.4.3 SBS energy consumption ($EC_{B_j}$)

According to [66], the EC of the SBS is given as

$$P_{comp} = P_{RFC} + P_{BCK} + P_{op_{u}} \quad (a)$$

$$P_{op_{u}} = \sum_{i=1}^{n} (EC_{op_{u}} + ET_{U_i, j}^{ul} + EC_{U_i, j}) \quad (b)$$

where $P_{comp}$ is the total energy consumption by the following: $P_{RFC}$, $P_{BCK}$, and $P_{op_{u}}$ are the energy consumption by radio frequency, backhaul link, and other operations related to the communication and contents, respectively. The serial number of the MS $U_{i,j}$ is represented as $EC_{op_{u}} + ET_{U_i, j}^{ul} + EC_{U_i, j}$, that denoted to the energy consumption of an SBS $B_j$ for executing operations, transmitting and receiving of contents, which are uploaded by the MS $U_{i,j}$, respectively. $ET_{U_i, j}^{ul}$ is the energy consumption for transmitting data by SBS $B_j$. $EC_{op_{u}}$ is the EC of an SBS $B_j$ for the execution of the cache replacement policy if the cache is full.

In case of a cache hit, $E_{i}$ is calculated only for the cost of receiving the MoTD at SBS $B_j$; otherwise $E_{i}$ is calculated for the cost of receiving the whole content.
Using Eqs (11) and (13), we can write;

\[ EC_{R_j} = P_{comp} + C_{EC} \] \hspace{1cm} (14)

Therefore, the average EC of the SBSs \((EC_{avg}^B)\) is given as

\[ EC_{avg}^B = \frac{\sum_{j=1}^{M} EC_{R_j}}{M} \] \hspace{1cm} (15)

Finally, according to (10) and (15), the EC of the overall cellular network is given by:

\[ EC_{N} = EC_{avg}^U + EC_{avg}^B \] \hspace{1cm} (16)

3.5 Spectrum Efficiency (SE) model

By definition, the uplink SE is determined by the relation between the uplink capacity \(R^{UL}\) and the total available bandwidth \(B\) as given by [67],

\[ SE = \frac{R^{UL}}{B} \] \hspace{1cm} (17)

According to the Shannon channel capacity, the maximum bit rate is given by

\[ R^{UL} = B \log_2(1 + SINR) \] \hspace{1cm} (18)

Then, according to (17) and (18), the SE is given as

\[ SE = \frac{R^{UL}}{B} = \frac{B \log_2(1 + SINR)}{B} = \log_2(1 + SINR) \] \hspace{1cm} (19)

4 Proposed framework for distributed uplink cache

In the distributed cache, there are three challenges. Firstly, content duplication among distributed cache. Secondly, an MS’s inability to know about the cache contents. Thirdly, the segmentation and caching distributively the new content with a large size. In order to address these challenges and improve the user’s quality of experience, a framework for distributed uplink caching have proposed. There are three main components of the proposed framework namely, generating a list of cache contents, performing matching at an MS, and content segmentation for distributed cache placement as shown in Fig 2.

4.1 Generating disparate list of distributed cache for uplink transmission

It is of paramount importance that an MS has a list of contents that are cached to avoid unnecessary uploads. In this subsection, the algorithm of generating the content list is explained in detail. An MBS is designated as a marker final content list (MFCL). The goal of MFCL is to create two content lists namely, Un-Duplicated content List (UDL) and Duplicated Content List (DCL). As its name implies, UDL consists of un-duplicated cache contents that can be used by an MS for content matching. Whereas, the DCL contains duplicated cache contents, which will be used for evicting identical contents. The rest of this section explains the process of creating these contents lists, which is also shown in Fig 3.
4.1.1 Processing of cache contents at MBS and SBSs. The processing involves creating a list of cached contents at an MBS and all SBSs in the network.

4.1.1.1 MBS cache contents. The information about the SBSs that are connected to an MBS can be expressed as:

\[ B_{\text{Loc}}^{M \times 4} = \begin{bmatrix} B_{L1}^D & C_{L1}^D & B_{L1}^L & C_{L1}^F \\ B_{L2}^D & C_{L2}^D & B_{L2}^L & C_{L2}^F \\ \vdots & \vdots & \vdots & \vdots \\ B_{LM}^D & C_{LM}^D & B_{LM}^L & C_{LM}^F \end{bmatrix} \]  

(20)
where $M$ rows present the total number of the SBSs. The columns present the characteristics of SBS such that $B^D_j$ and $C^D_j$ represent ID of an SBS $B_j$ and its cache, respectively. Whereas, $B^5_j$ is the SBS location in the coverage area of an MBS along with $C^5_j$, which represents the free space of each SBS and is computed as:

$$ C^S_j = S(C_{B_j}) - \sum_{j=1}^w S(C_{B_j,f}) $$

(21)

The list of contents of the MBS’s cache can be expressed as:

$$ C^M_C = \begin{bmatrix} P(G_{C_1,1}) & P(G_{C_1,2}) & \cdots & P(G_{C_1,\kappa}) \\ P(G_{C_2,1}) & P(G_{C_2,2}) & \cdots & P(G_{C_2,\kappa}) \\ \vdots & \vdots & \ddots & \vdots \\ P(G_{C_{\kappa},1}) & P(G_{C_{\kappa},2}) & \cdots & P(G_{C_{\kappa},\kappa}) \end{bmatrix}, $$

(22)

where $w$ rows present the total number of cached contents while the $\kappa$ is the maximum number of attributes of the content.

4.1.1.2 SBSs caches contents. The list of SBS’s cached contents that is denoted by $C^{w\times\kappa}_{B_j}$, and can be expressed as:

$$ C^{w\times\kappa}_{B_j} = \begin{bmatrix} P(C_{B_j,f^1,1}) & P(C_{B_j,f^1,2}) & \cdots & P(C_{B_j,f^1,\kappa}) \\ P(C_{B_j,f^2,1}) & P(C_{B_j,f^2,2}) & \cdots & P(C_{B_j,f^2,\kappa}) \\ \vdots & \vdots & \ddots & \vdots \\ P(C_{B_j,f^w,1}) & P(C_{B_j,f^w,2}) & \cdots & P(C_{B_j,f^w,\kappa}) \end{bmatrix}, $$

(23)

where $w$ is the total number of each SBS’s cached contents.

Then, the total number of contents in all the SBS’ caches can be represented as

$$ C_B = \{ C^{w\times\kappa}_{B_1}, C^{w\times\kappa}_{B_2}, \ldots, C^{w\times\kappa}_{B_M} \}. $$

(24)

4.1.1.3 Consolidated list of distributed cache contents. Every SBS sends its cache contents as shown in (23), to an MFCL, which is an MBS acting as a marker. After receiving the caches contents lists from all the SBSs, an MFCL applies row-wise combination function to generate a consolidated list $C^{T\times\kappa}_{D_j}$ as following:

$$ C^{T\times\kappa}_{D_j} = \begin{bmatrix} C^{w\times\kappa}_{G_C} \\ C^{w\times\kappa}_{B_1} \\ C^{w\times\kappa}_{B_2} \\ \vdots \\ C^{w\times\kappa}_{B_M} \end{bmatrix}, $$

(25)

where $T_j$ is the total number of all cached contents across a distributed cache.

4.1.2 Filtering similar contents. The consolidated list generated by an MFCL may have similar contents received from the various SBSs’ cache, in addition to different types of data such as nominal, ordinal, interval, and ratio. In order to remove duplication, the MFCL performs matching among attributes of contents of $C^{T\times\kappa}_{D_j}$ as shown in (25).
In order to compute dissimilarity in (25), we considered typical and target contents denoted by \(y\) and \(y^*\), respectively. The dissimilarity can be calculated as [68]

\[
\text{dissim}(y, y^*) = \frac{\sum_{j=1}^{\kappa} D_{yj}^{(p_j)} \cdot D_{yj}^{(p_j)*}}{\sum_{j=1}^{\kappa} D_{yj}^{(p_j)'}},
\]

(26)

where \(\text{dissim}(y, y^*)\) represents the difference between the typical content \(y\) and target content \(y^*\), \(\{p_j\}\) represents the attributes of content, \(\kappa\) is the maximum number of attributes of content. \(D_{yj}^{(p_j)}\) is the contribution of feature \(\{p_j\}\) to the dissimilarity between \(y\) and \(y^*\). Whereas, \(D_{yj}^{(p_j)*}\) is the indicator.

The fetching value principle of the \(D_{yj}^{(p_j)}\): If either \(D_{yj}^{(p_j)}\) or \(D_{yj}^{(p_j)*}\) has a missing value of the \(\{p_j\}\) attribute or \(D_{yj}^{(p_j)} = D_{yj}^{(p_j)*} = 0\) and attribute \(p_j\) is asymmetric, then \(D_{yj}^{(p_j)} = 0\); otherwise \(D_{yj}^{(p_j)} = 1\).

The fetching value principle of the \(D_{yj}^{(p_j)*}\):

1. If the attribute \(p_j\) of contents is numerical, interval or ratio type, then

\[
D_{yj}^{(p_j)} = \frac{|D_{yj}^{(p_j)} - D_{yj}^{(p_j)*}|}{\max_{y, y^*} D_{yj}^{(p_j)} - \min_{y, y^*} D_{yj}^{(p_j)*}},
\]

(27)

where \(h\) can take all the non-missing value of the attribute \(p_j\) of content.

2. If the attribute \(p_j\) of contents is nominal or binary, then

\[
D_{yj}^{(p_j)} = \begin{cases} 
0, & D_{yj}^{(p_j)} = D_{yj}^{(p_j)*} \\
1, & D_{yj}^{(p_j)} \neq D_{yj}^{(p_j)*}.
\end{cases}
\]

(28)

3. If the attribute \(p_j\) of contents is ordinal, then calculate ranking \(r_{yj}\), then

\[
z_{yj} = \frac{(r_{yj} - 1)}{(m_{p_j} - 1)},
\]

(29)

where \(r_{yj}\) represents the ranking of state in the attribute \(p_j\) of contents, \(m_{p_j}\) is the number of ordered states of \(p_j\), and treat \(z_{yj}\) as a numerical type.

The similarity between \(y\) and \(y^*\) is computed by:

\[
\text{sim}(y, y^*) = 1 - \text{dissim}(y, y^*),
\]

(30)

where \(\text{sim}(y, y^*)\) shows similarity of contents \((y, y^*)\).

The list of the values of similarity is denoted by \(\text{SIM}^{T \times \kappa}_{y, y^*}\) and can be expressed as:

\[
\text{SIM}^{T \times \kappa}_{y, y^*} = \begin{bmatrix}
1 & s_{1,2} & s_{1,3} & \ldots & s_{1,\kappa} \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
1 & s_{2,1} & 1 & \ldots & s_{2,\kappa} \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
s_{1,1} & 1 & s_{2,2} & \ldots & 1 \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
1 & s_{T_j,1} & s_{T_j,2} & \ldots & 1
\end{bmatrix}.
\]

(31)

The closest similar two contents is a column with a maximum value (value 1) among the values across each content. Alternatively, similarity can also be determined by setting a
threshold value \( \text{sim}^{th} \) such that

\[
\text{sim} = \begin{cases} 
\text{similar}, & \text{sim}(y, y^*) \geq \text{sim}^{th} \\
\text{dissimilar}, & \text{sim}(y, y^*) < \text{sim}^{th} 
\end{cases}.
\]

(32)

In this way, duplicate contents can be identified residing in different caches at SBSs, which can then be subsequently evicted to make space for new content.

### 4.1.3 Checking validity of cache contents.

After identifying similar contents, it is also important to identify the validity of the existing cache contents by the MFCL. The same content in the different cache should be evicted and kept only at one cache. To do this, we proposed Dynamic Validity Period (DVP) with the aim to compute the probability of either keeping or evicting the content from any of the distributed cache. It can be expressed as:

\[
DVP^{C_{k,j,l}} = RE_{B_j} + C_{l}^{B_j} + U_{j,l} + \varnothing^{C_{l}^{B_j}} + TimeEC_{C_{l}^{B_j}},
\]

(33)

where \( RE_{B_j} \) is the remaining energy of an SBS \( B_j \), \( C_{l}^{B_j} \) is the cache free space of an SBS as mentioned in (20) according to (21). \( U_{j,l} \) is the total number of the MSs served by the SBS \( B_j \). \( \varnothing^{C_{l}^{B_j}} \) is the popularity of each cached content according to (1). \( TimeEC_{C_{l}^{B_j}} \) is the remaining time of content in the cache and is given by:

\[
TimeEC_{C_{l}^{B_j}} = \text{Expire}_{time}^{C_{l}^{B_j}} - \text{Time}_{last}^{C_{l}^{B_j}},
\]

(34)

where \( \text{Expire}_{time}^{C_{l}^{B_j}} \) is the expire time of that content and the \( \text{Time}_{last}^{C_{l}^{B_j}} \) is the time of the last hit of that content.

The DVP is executed after performing the content similarity and determining the duplicate contents of the distributed cache. Let

\[
K = \max(DVP^{C_{1,j,l}}, \ldots, DVP^{C_{m,j,l}}),
\]

(35)

where \( 1 \leq m \leq M \). \( K \) represented the hash key of the contents of DVP \( DVP^{C_{k,j,l}} \). It means that represents all the attributes of the contents. That has highest validity. Therefore, it should be placed in UCL. All the remaining contents will be placed in DCL. This can formally be seen in (38) and (39).

### 4.1.4 Final list generation.

Now that the similarity and validity of the contents are determined, the final lists of duplicated and unduplicated contents can be generated. The UCL is the list of un-duplicate contents of the distributed cache, which can be used by an MS as a map to determine whether to upload contents or send a MoTD instead of uploaded the actual content. The UCL can be formed using (25) and (35). The value of \( K \) in (35) is used to select the content of \( C_{T_j,x} \) in (25) using the hash key \( H(C_{k,j,l}) \). \( K \) represents a row in \( C_{T_j,x} \) in (25), which has maximum validity (DVP). That is indicated by \( C_{T_j,x}^{C_{k,j,l}}(l, \ast) \), and can be presented as

\[
C_{T_j,x}^{C_{k,j,l}}(l, \ast) = |P(C_{k,j,l})| = P(C_{k,j,l}) \ldots P(C_{k,j,l}),
\]

(36)

where \( l \) is the serial number of the row with max DVP such that \( l \in [1, T_j] \).

\( C_{T_j,x}^{f, \ast} \) represents the row of the matrix in (25), which has maximum validity as shown in (35). Eq (36) shows max value of DVP of a single content. The same process is performed for all the contents of distributed cache. It will provide max values of DVP of all the duplicate
contents. It can be seen as a cluster of duplicated contents with different DVP values. The contents with maximum DVP value will be placed in UCL, while the remaining values will be placed in DCL. Based on this discussion, UCL can be represented as

\[ UCL^{T_{\text{max}}} = UCL^{T_{\text{max}}} + C_{D_{l}}^{T_{\text{max}}} (l, *), \]  

(37)

where \( l \in [1, T_l] \) represent the rows with max DVP values. When all the rows with max DVP values of contents are placed in UCL, the remaining duplicated contents are added to DCL and can be expressed as;

\[
UCL^{T_{\text{max}}} = \begin{bmatrix}
P_{c_{l,1}} & P_{c_{l,2}} & \cdots & P_{c_{l,x}} \\
P_{c_{l,1}} & P_{c_{l,2}} & \cdots & P_{c_{l,x}} \\
\vdots & \vdots & \ddots & \vdots \\
P_{c_{l,1}} & P_{c_{l,2}} & \cdots & P_{c_{l,x}} \\
\end{bmatrix},
\]

(38)

where \( T \leq T_l \) presents the total number of unduplicated contents among all distributed cache.

The DCL is the list of the duplicated contents of the distributed cache, which determines the contents to be evicted from SBS(s) caches. As previously mentioned that the max DVP values of contents are placed in UCL, the remaining duplicated contents are added to DCL and can be expressed as;

\[
DCL^{T_{\text{max}}} = \begin{bmatrix}
DVP^{\ell_{b_i}} & H(C_{(B_j/G_i^j)}) \\
DVP^{\ell_{b_i}} & H(C_{(B_j/G_i^j)}) \\
\vdots & \vdots \\
DVP^{\ell_{b_i}} & H(C_{(B_j/G_i^j)}) \\
\end{bmatrix},
\]

(39)

where \( T^* \) is the total number of the duplicated contents across a distributed cache to be eliminated from their caches.

**4.1.5 Duplication elimination at sbs and broadcast to the MSs.** After removing duplicate contents from MBS’s cache, UCL and DCL are sent to all the SBSs. The elimination of the duplicate contents as suggested by MFCL is the result of DVP in the DCL as shown in (39). The contents with low DVP are evicted from their caches and can be represented as victim(DVP^{\ell_{b_i}}, C_{b_i}, Time), where Time is the time of evicted content. The evicted contents are subsequently reported to MBS for updating B_{loc}^{M \times 4} as shown in (20).

Additionally, each SBS \( B_j \) broadcasts the UCL to all its serving MSs \( U_{j,k} \), which can be used for matching, subsequently. The contents of the distributed cache are updated continuously, and their popularity is getting changed for reasons such as times of uploads, downloads, shares, views, etc. Therefore, UCL is updated and broadcast to all the MSs at a periodic time to ensure content consistency among the distributed cache and the MSs.

**4.1.6 Proposed (DC)^2 scheme.** Based on the discussion, an algorithm is proposed, which is called Disparate Content of Distributed Cache (DC)^2 as shown in Algorithm 1. Each previous section from (4.1.1) to (4.1.5) corresponds to a step of the proposed algorithm. The steps of the proposed algorithm can be summarized as follows: The proposed (DC)^2 algorithm consists of 5 major steps. In step 1, all the caches at SBSs and MBS are processed followed by a similarity check in step 2. The proposed algorithm further checks the validity of the contents before generating the final lists in step 3 and 4, respectively. Finally, duplicate contents are removed from all the target SBSs and the un-duplicated list is sent to all the MSs to perform content matching in step 5.
Algorithm 1: Disparate Contents of Distributed Cache (DO)²

1. **Input:** \(B_{DO}, G_w, B, C_w = \{C_{w_1}, C_{w_2}, \ldots, C_{w_k}\}, U_j, i.\)
2. **Output:** UCL, DCL.
3. **Step 1:** Processing of cache contents at SBSs and MBS
   4. \(T, T', T'' = 0\), Counters of all, unduplicated and duplicated contents;
   5. \(Sim^{th} \) — the threshold of similarity;
   6. Create \(C_{DO}^{T'xx}\) by adding \(G_w\);
   7. for \((j = 1, j \leq M, j++)\) do
      8. for \((i = 1, i \leq \omega, i++)\) do
         9. Create new row in \(C_{DO}^{T'xx}\);
         10. Add \(C_{b,i}\) and its Attributes \(P(C_{b,i})\) to \(C_{DO}^{T'xx}\) (25);
         11. \(T_r++\);
   12. MFCL created \(C_{DO}^{T'xx}\) (25) ← list of all distributed cached contents;
4. **Step 2:** Filtering similar contents
   5. for \((y = 1, y' \leq T_r, y''++)\) do
      6. \(D_f\) ← Temporary list of duplicated contents;
      7. for \((y'' = y + 1, y' \leq T_r, y''++)\) do
         8. \(\text{dissim}(y, y') = 0\);
         9. Clear \(D^{(T_r + 1)}\);
         10. for \((q = 1, q \leq \kappa, q++)\) do
             11. if \((P_f || P_f' = \text{Null}) || (P_f || P_f' = 0)\) & \((P_f \text{ is Asymmetric})\) then
                 12. \(\delta^{(P_f)} = 0\)
                 13. else
                    14. \(\delta^{(P_f)} = 1\)
             15. if \((\text{Numeric — Interval — Ratio Attributes})\) then
                16. \(D_f\) according to (27);
                17. if \((\text{Nominal Attributes})\) then
                   18. \(D_f\) according to (28);
                   19. if \((\text{Ordinal Attributes})\) then
                      20. \(z_{p_j}\) according to (29);
                      21. Add \(D_f\) to \(D_f\);
                      22. Compute Dissimilarity \(\text{dissim}(y, y')\) according to (26);
                      23. \(\text{sim}(y, y')\) — Calculate Similarity according to (30);
                      24. Add \(\text{sim}(y, y')\) to SIMD (31);
5. **Step 3:** Checking validity of cache contents
   6. if \((\text{sim}(y, y') \leq Sim^{th})\) then
      7. \(DVP^{T'xx}\) — Compute DVP according to (33);
      8. Add \(D_f\) to \(D_f\);
      9. \(DVP^{T'xx}\) — Compute DVP according to (33);
      10. Add \(D_f\) to \(D_f\);
5. **Step 4:** Final List Generation
   11. Add \(\text{Max}(D_f)\) ⇒ UCL (38);
   12. \(T''++\);
   13. Add others \(D_f\) ⇒ DCL (39);
   14. \(T' = T'' + \text{Count}(D_f) - 1\);
   15. Clear \(D_f\);
6. **Step 5:** Duplication elimination at SBSs and Broadcast to the MSs.
   16. for \((i = 1, i \leq \omega, i++)\) do
      17. for \((j = 1, j \leq M, j++)\) do
         18. if \(H(DCL^{T'xx})_{j} \in (C_{w})\) then
            19. Add \(D_{f,j}\) to \(D_{f,j}\);
49. Remove \((C_{i,j'}, f)\) (as victim\((DVP_{i,j'}, C_{i,j}, Time))\);

50. Each SBS \(B_j\) broadcast UCL to all serving MSs \(U_j,i\).

51. Return UCL & DCL.

4.2 Cache assist uplink with an MS enabled matching

This section describes the mechanism of performing content matching at an MS after it receives UCL. The process of matching is performed in three steps as described below. In first step, an MS intends to upload new content to its serving SBS \(B_j\). It will make a consolidated list comprising of attributes of new contents and UCL. In second step, an MS will perform matching to check if its available in the cache. Lastly, the contents will only be uploaded in case cache miss i.e. unavailability of the contents in the cache. If the content is enlisted in UCL, an MS will not upload the content. Instead, MoTD will be sent to its serving SBS. It is worth mentioning that if matching is performed at an MS, it will significantly improve the spectrum and energy efficiency. The aforementioned steps are described in details as following.

4.2.1 Consolidated list of attributes of new content and UCL. This subsection provide details of constructing a consolidated list of attributes of new contents and the existing UCL. The main steps are as following.

1. Firstly, an MS creates a list comprising new content and its attributes as shown in (40).

\[
R_{j,i}^{\times K} = \begin{vmatrix}
P(R_{j,i}^1, 1) & P(R_{j,i}^1, 2) & \ldots & P(R_{j,i}^1, \kappa) \\
\vdots & \vdots & \ddots & \vdots \\
P(C_{j,i}, 1) & P(C_{j,i}, 2) & \ldots & P(C_{j,i}, \kappa) \\
\end{vmatrix}
\]

where \(P(R_{j,i}^1, \kappa)\) represents the new content attributes.

2. The unduplicated contents of the distributed cache is shown in (38), which is combined with (40) by using a row-wise combination function to generate a consolidated list. The consolidate list is denoted by \(O^{(T+1)\times K}\) and is given by;

\[
O^{(T+1)\times K} = \begin{vmatrix}
P(C_{1,1}) & P(C_{1,2}) & \ldots & P(C_{1,\kappa}) \\
P(C_{2,1}) & P(C_{2,2}) & \ldots & P(C_{2,\kappa}) \\
\vdots & \vdots & \ddots & \vdots \\
P(C_{T,1}) & P(C_{T,2}) & \ldots & P(C_{T,\kappa}) \\
P(R_{j,i}^1, 1) & P(R_{j,i}^1, 2) & \ldots & P(R_{j,i}^1, \kappa) \\
\end{vmatrix}
\]

The last row in (41) is the values of attributes of new content.

4.2.2 Matching the attributes of new content and UCL. After constructing the consolidated list, an MS will start matching to check whether the new content is available in the cache. Using (41), the matching among the attributes of new content and the contents in a distributed cache is performed by using the similar process as explained in Section (4.1.2), especially Eqs (26)–(30).

According to [69], the matching is performed between the control item and treat item. In this work, the control item is the UCL, while the new content is a target content. Then, the
matching will result in a dissimilarity matrix as shown in (42).

\[
D^{(T+1)\times K} = \begin{bmatrix}
0 & D(C_{j_1}, C_{j_2}) & 0 \\
D(C_{j_1}, C_{j_3}) & D(C_{j_2}, C_{j_3}) & 0 \\
\vdots & \vdots & \vdots \\
D(C_{j_1}, R_{j_3}^{\text{UCL}}) & D(C_{j_2}, R_{j_3}^{\text{UCL}}) & D(C_{j_3}, R_{j_3}^{\text{UCL}}) & 0
\end{bmatrix}
\]  

(42)

The last row of (42) shows the dissimilarity between the new and cached content, which can be used to either upload or discard the new contents.

4.2.3 Uploading dissimilar content and ignore similar content. After matching is performed, the content can either be uploaded or an MoTD is sent. If a match is found, the content is not uploaded and an MoTD is sent instead of the actual content. The content will be uploaded, if the match is not found. In other words, contents will only be uploaded in case of a cache miss.

4.2.4 Proposed UCMM scheme. Based on the discussion, an algorithm is proposed, which is called is called an Uplink Caching with Mobile Matching (UCMM) scheme as shown in Algorithm 2 and Fig 2-(Part-2). The proposed UCMM performs matching among the attributes of new contents and UCL at an MSs level to determine if new content is already in a cache, which will eliminate the need of uploading duplicate contents. Each previous section from (4.2.1) to (4.2.3) corresponds to a step of the proposed algorithm. The steps of the proposed algorithm can be summarized as follows: In step 1, consolidate the lists of the attributes of new contents and UCL at a list followed by a similarity check in step 2. Finally, decides whether to upload dissimilar content or ignore similar content by sending a MoTD to the serving SBS in step 3.

**Algorithm 2: Uplink Caching with Mobile Matching (UCMM)**

1. **Input:** \( UCL^{(T_1)\times K}, R_{i}^{\text{UCL}} \).
2. **Output:** Matching factor \( \text{Matching}_R \), Miss/Hit factor \( \text{Miss}_n \) or \( \text{Hit}_n \).
3. \( \text{Miss}_n \leftarrow 0 \), \( \text{Hit}_n \leftarrow 0 \), \( \text{Matching}_R \leftarrow \text{false} \), \( \text{sim}_n \leftarrow \text{Value of threshold of Similarity} \); \( D^{(T+1)\times K} \leftarrow \text{Dissimilarity matrix(42)} \).
4. **Step-1:** Consolidated List of Attributes of New Content and UCL.
5. Create \( R_{i}^{\text{UCL}} \).
6. Get \( UCL^{(T_1)\times K} \).
7. Create \( O^{(T_1)\times K} \) refers(41) by consolidated (38 and 40).
8. **Step-2:** Matching the Attributes of New Content and UCL.
9. for \((y = 1, y \leq (T + 1), y++)\) do
10. \( \text{dissim}(C_{j_1}, R_{j_2}^{\text{UCL}}) \leftarrow 0; \)
11. Clear \( D^{(T+1)\times K} \);
12. **for** \((q = 1, q \leq x, q++)\) do
13. **if** \((P_{j_1}^{(P)} \parallel P_{j_2}^{(P)} = \text{Null}) \parallel (P_{j_1}^{(P)} \parallel P_{j_2}^{(P)} = 0) \& (P_{j_1} \text{ is Asymmetric})\) then
14. \( \delta_{j_1,j_2} = 0 \)
15. else
16. \( \delta_{j_1,j_2} = 1 \)
17. **if** (Numeric — Interval — Ratio Attributes) then
18. \( D_{j_1}^{(P)} \leftarrow \text{according to (27)}; \)
19. **if** (Nominal Attributes) then
20. \( D_{j_1}^{(P)} \leftarrow \text{according to (28)}; \)
4.3 Content segmentation and distributed placement

In the case of distributed caching, the size of contents is a major limiting factor towards the cache effectiveness [27, 38, 56]. The new content size may be too large to accommodate in a single cache, efficiently. Therefore, it is recommended that the new content should be split into smaller segments with same size to be stored distributively. In this vein, we proposed a scheme that advocates to divide the new content into different segments as a function of new content size \( S(R_{j_i}^t) \) and available cache space of the corresponding and its neighboring SBSs as shown in Fig 2-(Part-3) and (Fig 4). The rationale of this content segmentation is the effective placement due to the smaller size in distributed cache across multiple SBSs. According to [56], the \( Q \) is equal \( Q \leq (1 + \text{count}(B_j)) \), where \( 1 \) is the local SBS \( B_j \) and \( \text{count}(B_j) \) is the number of neighboring SBSs, which have a non-void intersection with \( B_j \) and enough free space to cache at least a segment as \( B_j \triangleq \{ B_j \in B_{\text{loc}}^M : C_{\text{loc}}^{B_j} \geq S(R_{j_i}^t) \} \).

In order to facilitate its storage distributively, the segment(s) will be cached in a target distributed cache based on its free space according to (20) and (21). An MDS code method is used to encode the new content into packets as mentioned in Section (3.2.2).

The result of the above discussions is programmatically extracted in a new proposed scheme, which is called Content Segmentation with Distributed Placement (CSDP) as shown in Algorithm-3, and in Fig 2-(part-3). The CSDP consists of 2 major steps, where step 1 performs preparation of the required information based on the size and hash key of the new content. In addition, free space of the local and its neighboring SBSs is also considered. Step 2, corresponds to new content splitting into \( Q \) segments, encoding, and cache placement of segmented contents in a distributed manner.

**Algorithm 3:** Content Segmentation with Distributed Placement (CSDP) Scheme

1. **Input:** \( H(R_{j_i}^t), S(R_{j_i}^t), SBS_j, R_{b_j} \).
2. **Output:** \( E_0 \).
3. **Step 1:** Preparing the require information.
   4. **Step 2:** Computing Similarity \( \text{sim}(C_{R_{j_i}^t}, R_{j_i}^t) \) according to (26).
   5. **Step 3:** Computing Dissimilarity \( \text{dissim}(C_{R_{j_i}^t}, R_{j_i}^t) \) according to (29).
   6. **Step 4:** Select \( Q \) segments based on \( \text{sim}(C_{R_{j_i}^t}, R_{j_i}^t) \) and \( \text{dissim}(C_{R_{j_i}^t}, R_{j_i}^t) \) according to (20) and (21).
   7. **Step 5:** Encoding \( Q \) segments into packets.
   8. **Step 6:** Placing \( Q \) segments into distributed cache.
5 $H(R_i^j)$: The hash key of the new content;
6 Get the free space of the target SBSs according to (20 and 21);
7 $B^s \triangleq \{B_j \in B^{MC} : C_{s,j}^i > S(R_i^j)\}$ // Set the target SBSs;
8 **Step 2**: Split new content and cached it distributively
9 $Q = 1 + \text{count}(B_j)$ // total number of segments;
10 for $(q = 1, q = Q, q++)$ do
11 Check the $C_{s,j}^i$ according to (20 and 21);
12 if $(q = 1) \land (C_{s,j}^i > S(R_i^j))$ then
13 Store Seg$_q$ or set of Seg$_q$s in $B_j$;
14 Encode $e_{q,j}$ using MDS code, $e_{q,j} \rightarrow e_{q,j}^B$;
15 else
16 Select $B_j$ from $B^s$;
17 Check $C_{s,j}^i$ of neighboring SBSs according to (20 and 21);
18 if $(C_{s,j}^i > S(R_i^j))$ then
19 Store Seg$_q$ or set of Seg$_q$s in $B_j$;
20 Encode $e_{q,j}$ using MDS code, $e_{q,j} \rightarrow e_{q,j}^B$;
21 Create the map of the set encoded packets: $E_q = \{e_1, e_2, ..., e_q\}$;
22 Change the status of new content to be a cached content;
23 Return $E_q$.
4.4 Complexity analysis of the proposed framework

As mentioned, and discussed in the previous sections, \( M \) is the total number of distributed cache. Each cache store \( w \) contents with a total of \( T_f \) contents in the whole distributed network. Each content has \( \kappa \) attributes. Additionally, the unduplicated and duplicated contents among distributed cache are \( T \), and \( T^* \), respectively. As well, the proposed framework consists of three parts as follows: (CD\(^2\), UCMM, and CSDP).

The overall complexity of the proposed framework is \( (O(M.w)^2) + O(M.w) + O(M)) \), which is computed for the three parts as follows:

4.4.1 Complexity of CD\(^2\) algorithm. The CD\(^2\) is performed to solve the issue of eliminating the duplicated contents among the distributed cache as shown in algorithm 1, which is executed by many operations. The main operations of the iterative have complexity as the following:

1. The operation of the processing of the cache contents at the distributed cache has complexity \( O(M.w) \).
2. The operation of filtering similar contents by performing the matching among the attributes of the contents has complexity \( O(T_f^2) \).
3. The eliminating of the duplicate contents from the target distributed cache has complexity \( O(T^*) \).
4. Finally, the overall complexity of the CD\(^2\) is \( O(M.w) + O(T_f^2) + O(T^*) \), where \( T_f = (M.w) \) but \( O(T_f^2) > O(T^*) \) and \( O(T_f^2) > O(m.w) \). Then, the overall complexity of the proposed framework is \( O(T_f^2) = O((M.w)^2) \).

4.4.2 Complexity of UCMM algorithm. The solution to the cache uplink problem without duplication of the mobile data to be uploaded is solved by algorithm 2 called UCMM. The overall complexity of UCMM is \( O(1) \) in the best case, while \( O(M.w) \) in the worst case.

4.4.3 Complexity of CSDP algorithm. The solution of the segmentation of the new content with a large size and cache distributively is solved by algorithm 3 called SCDP. The new content will be segmented into \( Q \) segments with a smaller size to be cached distributively in the target caches, which is equal to or smaller than \( M \). The segmentation of new content has complexity \( O(Q) \). While the caching segments of each content have complexity \( O(M) \). Therefore, the overall complexity of the CSDP is \( O(M) \).

5 Experimental design and evaluation

We have compared our proposed framework in different scenarios such as uplink with no-caching (No-cache), cache assisted uplink (Each-cache) [33] and uplink with collaborative distribution caching (SBS-CoDc) [38]. As its name implies in a no-caching scenario, we have considered the absence of caching at the SBS, while in the cache-assisted uplink, we consider uplink with the support of cache. The simulation parameters are shown in Table 3.

5.1 Performance evaluation metrics

In our system model, \( EC_{avU} \), \( EC_{avB} \), and \( EC_N \) denote the average energy consumption of the MSs, and SBSs, and the overall network, respectively. In addition, \( UL_{avU} \), \( UL_{avB} \), and \( UL_{avNet} \) denote the uplink data rate of an MS, an SBS, and the overall network, respectively. For a comprehensive
evaluation of the performance of the proposed framework, it is compared with the existing schemes in [33, 38]. The following metrics are used, which are computed as (43)–(55).

5.1.1 Measurements cache availability and efficiency. According to (3) and (4), the cache hit ratio of the overall network is given as

$$\text{Net}_{\text{cache}} = \frac{\sum_{n=1}^{M} \text{Hit}_{B_n}}{\sum_{n=1}^{M} \text{Hit}_{B_n} + \sum_{n=1}^{M} \text{Miss}_{B_n}} = \sum_{j=1}^{M} \frac{\text{Hit}_{C_j}}{\text{Hit}_{C_j} + \text{Miss}_{C_j}},$$  \hspace{1cm} (43)$$

where $\text{Hit}_{B_n}$ is a cache hit ratio of an SBS $B_n$ and $\text{Miss}_{B_n}$ is the cache miss ratio of the same SBS.

While the cache miss ratio of the overall network is given as

$$\text{Net}_{\text{cache}} = (1 - \text{Net}_{\text{cache}}),$$  \hspace{1cm} (44)$$

5.1.2 Throughput. Throughput ($\text{TH}$) is the ratio of the average number of successfully transmitted data in GB per second [70] and is computed as follows:

The throughput of MS ($\text{TH}_i$) is given as,

$$\text{TH}_i = \frac{T_{\text{Num}}}{T},$$ \hspace{1cm} (45)$$

where $T_{\text{Num}}$ is the total number of successfully transmitted data and $T$ is the transmit time.

The average throughput of MSs ($\text{TH}_{\text{avg}}$) is given as,

$$\text{TH}_{\text{avg}} = \frac{\sum_{i=1}^{N} \text{TH}_i}{N},$$ \hspace{1cm} (46)$$
The average throughput SBSs ($\text{TH}^{\text{avg}}_B$) is given as,

$$\text{TH}^{\text{avg}}_B = \frac{\sum_{j=1}^{M} \text{TH}_j}{M}$$  \hspace{1cm} (47)

5.1.3 Energy efficiency measurements. The Energy Efficiency (EE) of the uplink is the ratio of uplink data rate to the energy consumption, where the unit is GB/J/s. Area Energy Efficiency (AEE) is measured based on both the energy consumption and the size of the coverage area, where the unit is GB/Joule per area (Km$^2$) [66, 67, 71].

In this regard, the average EE of the MSs ($\text{EE}^U$) can be calculated as,

$$\text{EE}^U = \frac{\sum_{i=1}^{N} (R^{UL}_{U,j} / E_{C_{U,j}})}{N}$$  \hspace{1cm} (48)

The average EE of SBSs ($\text{EE}^B$) can be calculated as,

$$\text{EE}^B = \frac{\sum_{j=1}^{M} (R^{UL}_{B,j} / E_{C_{B,j}})}{M}$$  \hspace{1cm} (49)

Furthermore, the network AEE is computed in two ways. Firstly, based on the uplink data rate and energy consumption, which is denoted by $(\text{AEE}^\text{UL}_N)$ and can be calculated as,

$$\text{AEE}^\text{UL}_N = \frac{R^{UL}_{\text{Net}}}{E_{C_{N}}}$$  \hspace{1cm} (50)

Secondly, based on average EE of MSs and SBSs in addition to the coverage size, which is denoted by $(\text{AEE}^\text{Size}_N)$ and can be calculated as,

$$\text{AEE}^\text{Size}_N = \frac{\sum_{j=1}^{M} \sum_{i=1}^{N} \text{EE}_U + \sum_{j=1}^{M} \text{EE}_B}{\sum_{j=1}^{M} S_{B_j}}$$  \hspace{1cm} (51)

5.1.4 Spectral efficiency measurements. Spectral Efficiency (SE) is defined as the uplink data rate per bandwidth measured in GB/s/Hz, which is an important metric to represent the performance of radio resource utilization of the network as given by [67, 72].

According to (19), the average SE for MSs ($\text{SE}^{\text{avg}}_U$) can be calculated as following

$$\text{SE}^{\text{avg}}_U = \frac{\log_2 \left(1 + \frac{TP_u^{UL} H^{ul}_{U,j} \|d(U_{j,i}, B_j)\|^{-\alpha}}{\sum_{i=1}^{N} + \sigma^2} \right)}{N}$$  \hspace{1cm} (52)

Similarly, the average SE of SBSs ($\text{SE}^{\text{avg}}_B$) can be calculated as following

$$\text{SE}^{\text{avg}}_B = \frac{\log_2 \left(1 + \frac{TP_u^{UL} H^{ul}_{B,j} \|d_{B_j,G}\|^{-\alpha}}{\sigma^2} \right)}{M}$$  \hspace{1cm} (53)

In addition to, Area Spectral Efficiency (ASE) ($\text{ASE}_{\text{SE}}$) (GB/Hz per area Km$^2$) can be calculated as following

$$\text{ASE}_{\text{SE}} = \frac{\sum_{j=1}^{M} \text{SE}_{B_j} + \sum_{i=1}^{N} \text{SE}_{U,i}}{M}$$  \hspace{1cm} (54)

5.1.5 Overall Cache Efficiency (OCE) measurements. The OCE is the ratio of the cumulative cache hits to cumulative demands. That reflects the overall number of cache hits until a
specific time (time slot $t$). An overall cache Efficiency (OCE) is denoted by $O_{CE}$, and is given according to [73] as,

$$OCE = \sum_{i=1}^{M} \frac{Hit_{t}^{C_{B_{i}}}}{S_{D}}$$  \hspace{1cm} (55)$$

where, $\sum_{i=1}^{M} Hit_{t}^{C_{B_{i}}}$ is the cumulative distributed cache hit ratio, while $S_{D}$ is cumulative demands.

6 Numerical results

The simulation is performed considering the proposed framework and the existing models namely, No-Cache, Each-cache [33] and SBS-CoDc [38]. In addition, their sub-models in different scenarios with the same simulation parameters are listed in Table 3. The final result is validated by the different performance metrics, which are presented in (5.1) and shown below in different Figures.

6.1 Cache hit and miss ratio

Fig 5, shows the average cache hit and miss ratio of the Each-cache, and SBS-CoDc along with our Proposed Framework according to (43) and (44).

We can see a slight increase in the average cache hit ratio of SBS-CoDc as compared to Each-cache because of the lack of cooperation among SBSs. The cache hit ratio of Each-Cache is computed separately and summed subsequently. Our proposed framework outperforms
SBS-CoDc by improving the cache hit ratio by 9%. The main reason is the effective use of the CSDP scheme, which improves the cache hit ratio. We can also see that our proposed framework has significantly reduced the cache miss ratio.

The rationale is the distribution of contents among different SBSs along with the UCL, which acts as a map for MSs to efficiently locate the cached contents. Due to these improvements, the traffic load at the access network, as well as backhaul link, is significantly reduced, which subsequently improves EE and SE.

6.2 Average Energy Consumption (EC)

The average energy consumption of MSs and SBSs is shown in Figs 6 and 7, respectively. Our proposed framework is compared with No-Cache, Each-cache, and SBS-CoDc. The percentage improvements of average energy consumption of MSs and SBSs are shown in Table 4.

We can see that our proposed framework performs significantly better as compared to No-Cache and Each-Cache scenarios. However, about 17% and 13% improvements are noticed as compared to SBS-CoDc, which also has inherent advantages of distributed scenarios. However, our proposed scheme uses UCL for matching at an MS. As the cache hit ratio of our proposed scheme is improved, it also positively affects the average energy consumption by limiting the amount of unnecessary data upload.

6.3 Improvement of uplink throughput

The average Uplink Throughput (TH) of our proposed framework is compared with the existing schemes as shown in Figs 8 and 9 of the MSs and SBSs, respectively. The percentage improvements are shown in Table 5.
We can see that the THs of our proposed framework for both MS and SBS are significantly better than the No-Cache scheme for obvious reasons. Similarly, compared to Each-Cache, our proposed framework performs better due to its distributed nature. The result shows that our framework is better than SBS-CoDc because of the effective use of MoTD rather than uploading new content. Furthermore, as compared to the existing schemes, matching is done at an MS rather than at an SBS, which means no content upload that subsequently improves TH.

6.4 Improvement in energy efficiency

Energy efficiency (EE) is an effective way to show performance improvement. As one can predict that the lower energy consumption as shown previously, will also entail the EE. Our proposed framework is evaluated based on EE and compared with the existing schemes. The EE of MS and SBS increased with the increasing number of MSs is shown in Figs 10 and 11. All the percentage improvements of the proposed framework as a result of comparison with existing schemes are shown in Table 6.

Table 4. Percentage improvement of average EC of MSs and SBSs.

| Existing Scheme | MS      | SBSs | Based on |
|-----------------|---------|------|----------|
| No-Cache        | 38.71%  | 29.46% | (9), (10), (14), (15) |
| Each-Cache      | 29.63%  | 20.4%  | -do-     |
| SBS-CoDc        | 17.39%  | 12.71% | -do-     |

Fig 7. Comparison average energy consumption of SBSs.
https://doi.org/10.1371/journal.pone.0268294.g007
Fig 8. Comparison average throughput of MSs.  
https://doi.org/10.1371/journal.pone.0268294.g008

Fig 9. Comparison average throughput of SBSs.  
https://doi.org/10.1371/journal.pone.0268294.g009
We can see in Fig 10, that the average EE of our proposed framework is better as compared to existing schemes. Compared to SBS-CoDc, our proposed framework improves the average EE by 41%. The rationale is the use of UCL for matching at the MS, which effectively reduces the contents upload in case of a cache hit ratio. Similarly, Fig 11, shows the improved average EE of SBSs of our proposed framework. We can see the average EE of our proposed framework as compared to other schemes with an increasing number of MSs. Our proposed framework outperforms existing schemes by significantly improving average EE by 46% as compared to SBS-CODc. The main reason is the improved hit ratio of our proposed framework, which implicitly improves the average EE because of the use of MoTD that represents the new contents.

We have also used the metric of Area Energy Efficiency (AEE) to show the performance improvement of our proposed scheme. We computed AEE using two ways, firstly as a function of uplink data and secondly as a function of EE. Figs 12 and 13, show the AEE of No-cache, Each-cache, SBS-CoDc, and proposed framework of different numbers of MSs based on two mentioned ways, respectively.

| Existing Scheme | MS   | SBS  | Based on       |
|-----------------|------|------|----------------|
| No-Cache        | 90.04% | 88.24% | (46) and (47) |
| Each-Cache      | 48.88% | 52.38% | -do-          |
| SBS-CoDc        | 40.00% | 28.00% | -do-          |

Table 5. Percentage improvement of average throughput of MSs and SBSs.
Fig 12, shows the AEE performance as a function of the uplink data rate, which is computed by dividing the total uplink data rate by the total energy consumption per unit area based on (50). Fig 12, shows that our proposed framework has increased the AEE by 2.34 GB/J/km² as compared to SBS-CoDC. A summary of the result of the comparison with other schemes is shown in Table 7.

Furthermore, Fig 13, shows AEE performance as a function of the cell size, which is computed by dividing the total energy efficiency by the total size of the network based on (51) in order to assess the EE of the overall network to its size. In Fig 13, we can see our proposed framework has increased the average AEE by 43% as compared to SBS-CoDC. These improvements are credited to the improved hit ratio, higher TH, and better EE.

### 6.5 Spectral efficiency

The average spectral efficiency (SE) of MSs as compared to No-cache, Each-cache, SBS-CoDC, and our proposed framework for different numbers of MSs is shown in Fig 14. We can see that our proposed framework has improved the SE almost by 16% as compared to SBS-CoDC. Whereas, compared to Each-Cache, improvement is almost 37%. Similarly, Fig 15, shows the

| Existing Scheme  | MS   | SBSs  | Based on       |
|------------------|------|-------|----------------|
| No-Cache         | 180.04% | 165.68% | (48) and (49)  |
| Each-Cache       | 97.75%  | 91.46%  | -do-           |
| SBS-CoDC         | 41.28%  | 46.18%  | -do-           |

https://doi.org/10.1371/journal.pone.0268294.t006
Fig 12. Comparison AEE based on uplink data rate and EC of the overall network.

Fig 13. Comparison AEE based on coverage area and EE.

https://doi.org/10.1371/journal.pone.0268294.g012

https://doi.org/10.1371/journal.pone.0268294.g013
average SE of SBSs of existing schemes as compared to our proposed framework. The summary of improvement is shown in Table 8.

The rationale of improved SE is a significant reduction in the number of uplink contents. Since matching is done at an MS that facilitates the decision of either uploading the contents or not. In case of a cache hit, the contents are not uploaded and hence the bandwidth is saved for the use of other requests of the remaining MSs. In this way, a significant amount of spectrum can be saved and more requests can be entertained, which ultimately improves the SE.

Fig 16, shows the ASE of No-cache, Each-cache, SBS-CoDc, and proposed framework for different numbers of MSs. We can see in Fig 16, that our proposed framework has improved the ASE by 24% as compared to SBS-CoDc.

### 6.6 Improvement of overall distributed cache efficiency

The overall cache efficiency (OCE) of the distributed cache as compared to Each-cache, SBS-CoDc, and our proposed framework for different numbers of MSs is shown in Fig 17.

Fig 17, shows the OCE of the distributed cache of existing schemes as compared to our proposed framework. We can see that our proposed framework has improved the OCE almost by

---

**Table 7. Percentage improvement of average AEE.**

| Existing Scheme | Uplink Data | EE     | Based on |
|-----------------|-------------|--------|----------|
| No-Cache        | 172.44%     | 172.82% | (50) and (51) |
| Each-Cache      | 94.03%      | 94.62%  | -do-     |
| SBS-CoDc        | 43.35%      | 43.64%  | -do-     |

https://doi.org/10.1371/journal.pone.0268294.t007

---

![Average Spectral Efficiency of Mobiles](https://doi.org/10.1371/journal.pone.0268294.g014)

**Fig 14. Comparison average spectral efficiency of MSs.**

https://doi.org/10.1371/journal.pone.0268294.g014
28%, and 7.41% as compared to Each-cache, and SBS-CoDc, respectively. This is because the contents of the distributed cache are available to all the MSs, irrespective of their serving SBSs. In addition, the distributed cache miss ratio is less, which improves the cache efficiency and reduces access time to cache.

7 Conclusion

This paper proposed an efficient uplink cache framework based on a distributed scenario. The proposed framework leveraged the content matching at an MS in contrast to the existing schemes, which perform it at an SBS. In addition, the content matching at an MS has significantly improved the energy and spectral efficiency. The rationale is the reduced uplink contents due to local content matching at an MS. Furthermore, the proposed framework is based on the effective distribution of cache contents over cooperative SBSs, which improves the cache hit ratio. This entailed the subsequent improvement in throughput, energy consumption, and spectrum for MSs as well as SBSs. Our analysis shows that our proposed framework improves the EE and SE of the access network by 41.28% and 15.58%, respectively. Furthermore, an increase of 46.18% and 28.00% is respectively calculated as EE and SE for the

![Average Spectral Efficiency of SBSs](https://doi.org/10.1371/journal.pone.0268294.g015)

| Existing Scheme | MS     | SBSs   | Based on     |
|-----------------|--------|--------|--------------|
| No-Cache        | 67.92% | 88.24% | (52) and (53)|
| Each-Cache      | 36.92% | 52.38% | -do-         |
| SBS-CoDc        | 15.58% | 28.00% | -do-         |

https://doi.org/10.1371/journal.pone.0268294.t008
Fig 16. Comparison area spectral efficiency (ASE).

https://doi.org/10.1371/journal.pone.0268294.g016

Fig 17. Comparison overall distributed cache efficiency.

https://doi.org/10.1371/journal.pone.0268294.g017
backhaul link. As well as, the OCE of the proposed framework improves by 28%, and 7.41%, as compared to Each-Cache, and SBS-CoDc, respectively.

Supporting information
S1 File.
(ZIP)
S2 File.
(ZIP)
S3 File.
(ZIP)
S4 File.
(ZIP)

Author Contributions
Conceptualization: Mubarak Mohammed Al Ezzi Sufyan, Waheed Ur Rehman.
Data curation: Mubarak Mohammed Al Ezzi Sufyan, Waheed Ur Rehman, Qazi Ejaz Ali.
Formal analysis: Mubarak Mohammed Al Ezzi Sufyan, Abdul Rahman Al-Salehi, Abdul Haseeb Malik.
Investigation: Mubarak Mohammed Al Ezzi Sufyan, Waheed Ur Rehman.
Methodology: Mubarak Mohammed Al Ezzi Sufyan, Tabinda Salam.
Project administration: Waheed Ur Rehman.
Resources: Mubarak Mohammed Al Ezzi Sufyan.
Software: Mubarak Mohammed Al Ezzi Sufyan.
Supervision: Waheed Ur Rehman.
Validation: Mubarak Mohammed Al Ezzi Sufyan, Waheed Ur Rehman, Tabinda Salam.
Visualization: Mubarak Mohammed Al Ezzi Sufyan.
Writing – original draft: Mubarak Mohammed Al Ezzi Sufyan.
Writing – review & editing: Mubarak Mohammed Al Ezzi Sufyan, Waheed Ur Rehman, Tabinda Salam, Abdul Rahman Al-Salehi.

References
1. Cisco Annual Internet Report (2018-2023), White Paper, Cisco Public.
2. Ge X., Yang B., Ye J., Mao G., Wang C. and Han T., “Spatial Spectrum and Energy Efficiency of Random Cellular Networks,” in IEEE Transactions on Communications, vol. 63, no. 3, pp. 1019–1030, March 2015. https://doi.org/10.1109/TCOMM.2015.2394386
3. M. Katz, P. Pirinen and H. Posti, “Towards 6G: Getting Ready for the Next Decade,” 2019 16th International Symposium on Wireless Communication Systems (ISWCS), 2019, pp. 714-718.
4. Mattera D. and Tanda M. J. P. C., “Windowed OFDM for small-cell 5G uplink,” Physical Communication, vol. 39, p. 100993, 2020. https://doi.org/10.1016/j.phycom.2019.100993
5. Chen L., Huang W., Deng D., Xia J., Chen B., and Zhu F., “Multi-antenna processing based cache-aided relaying networks for B5G communications,” Physical Communication, p. 101141, 2020. https://doi.org/10.1016/j.phycom.2020.101141
6. Xia J. et al., “Cache-aided mobile edge computing for 5G wireless communication networks,” EURASIP Journal on Wireless Communications and Networking, vol. 2020, no. 1, p. 15, 2020. https://doi.org/10.1186/s13638-019-1612-0

7. Salam T., Rehman W. U. and Tao X., “Data Aggregation in Massive Machine Type Communication: Challenges and Solutions,” in IEEE Access, vol. 7, pp. 41921–41946, 2019. https://doi.org/10.1109/ACCESS.2019.2906880

8. Rehman W. U., Salam T., Almogren A., Haseeb K., Ud Din I. and Bouk S. H., “Improved Resource Allocation in 5G MTC Networks,” in IEEE Access, vol. 8, pp. 49187–49197, 2020. https://doi.org/10.1109/ACCESS.2020.2974632

9. B. Serbetci and J. Goseling, “On Optimal Geographical Caching in Heterogeneous Cellular Networks,” 2017 IEEE Wireless Communications and Networking Conference (WCNC), 2017, pp. 1-6.

10. Y. Hua, “Distributed caching in the future generation networks,” Loughborough University, 2020.

11. Salam T., Tao X., Chen Y., and Zhang P., “A trust framework based smart aggregation for machine type communication,” Science China Information Sciences, vol. 60, no. 10, pp. 1–15, 2017. https://doi.org/10.1007/s11432-017-9186-6

12. H. Elshaer, F. Boccardi, M. Dohler and R. Irmer, “Downlink and Uplink Decoupling: A disruptive architectural design for 5G networks,” 2014 IEEE Global Communications Conference, 2014, pp. 1798-1803.

13. Li L., Zhao G. and Blum R. S., “A Survey of Caching Techniques in Cellular Networks: Research Issues and Challenges in Content Placement and Delivery Strategies,” in IEEE Communications Surveys & Tutorials, vol. 20, no. 3, pp. 1710–1732, thirdquarter 2018. https://doi.org/10.1109/COMST.2018.2820021

14. Salam T., Rehman W. U. and Tao X., “Cooperative Data Aggregation and Dynamic Resource Allocation for Massive Machine Type Communication,” in IEEE Access, vol. 6, pp. 4145–4158, 2018. https://doi.org/10.1109/ACCESS.2018.2791577

15. Y. Xu, Z. Wang, Y. Li, T. Lin, W. An and S. Ci, “Minimizing Bandwidth Cost of CCN: A Coordinated In-Network Caching Approach,” 2015 24th International Conference on Computer Communication and Networks (ICCCN), 2015, pp. 1-7.

16. Wu X., Li Q., Leung V. C. M. and Ching P. C., “Joint Fronthaul Multicast and Cooperative Beamforming for Cache-Enabled Cloud-Based Small Cell Networks: An MDS Codes-Aided Approach,” in IEEE Transactions on Wireless Communications, vol. 18, no. 10, pp. 4970–4982, Oct. 2019. https://doi.org/10.1109/TWC.2019.2931335

17. Wu X., Li Q., Li X., Leung V. C. M. and Ching P. C., “Joint Long-Term Cache Updating and Short-Term Content Delivery in Cloud-Based Small Cell Networks,” in IEEE Transactions on Communications, vol. 68, no. 5, pp. 3173–3186, May 2020. https://doi.org/10.1109/TCOMM.2020.2979086

18. Sun Y., Zhu Z. and Fan Z., “Distributed Caching in Wireless Cellular Networks Incorporating Parallel Processing,” in IEEE Internet Computing, vol. 22, no. 1, pp. 52–61, Jan./Feb. 2018.

19. Hou R., Cai J., and Lui K.-S., “Distributed cache-aware CoMP transmission scheme in dense small cell networks with limited backhaul,” Computer Communications, vol. 138, pp. 11–19, 2019. https://doi.org/10.1016/j.comcom.2018.12.004

20. A. Sadeghi, A. G. Marques and G. B. Giannakis, “Distributed Network Caching via Dynamic Programming,” ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 4574-4578.

21. P. Ostovari, J. Wu, and A. Kheirshah, “Efficient online collaborative caching in cellular networks with multiple base stations,” in 2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 2016: IEEE, pp. 136-144.

22. Lu B., Zhu F., Xia J., Li X., Zhou W., and Fan L. J. P. C., “Interference suppression by exploiting wireless cache in relaying networks for 5G communications,” Physical Communication, vol. 42, p. 101162, 2020. https://doi.org/10.1016/j.phycom.2020.101162

23. Tamoor-ul-Hassan S., Bennis M., Nardelli P. H. J. and Latva-aho M., “Caching in Wireless Small Cell Networks: A Storage-Bandwidth Tradeoff,” in IEEE Communications Letters, vol. 20, no. 6, pp. 1175–1178, June 2016. https://doi.org/10.1109/LCOMM.2016.2543698

24. Kuang S. and Liu N., “Analysis and Optimization of Random Caching in Multi-Antenna Small-Cell Networks With Limited Backhaul,” IEEE Transactions on Vehicular Technology, vol. 68, no. 8, pp. 7789–7803, 2019. https://doi.org/10.1109/TVT.2019.2924642

25. M. Karaliopoulos, L. Chatziileftheriou, G. Darzanos and I. Koutsopoulos, “On the Joint Content Caching and User Association Problem in Small Cell Networks,” 2020 IEEE International Conference on Communications Workshops (ICC Workshops), 2020, pp. 1-6.
26. K. Tokunaga, K. Kawamura, and N. Takaya, “High-speed uploading architecture using distributed edge servers on multi-RAT heterogeneous networks,” in 2016 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN), 2016: IEEE, pp. 1-2.

27. Q. Jia, R. Xie, T. Huang, J. Liu and Y. Liu, “Energy-efficient cooperative coded caching for heterogeneous small cell networks,” 2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2017, pp. 468-473.

28. Z. Zhang, Z. Chen, and B. Xia, “Cache-enabled uplink transmission in wireless small cell networks,” in 2018 IEEE International Conference on Communications (ICC), 2018: IEEE, pp. 1-6.

29. Papazafeiropoulos A. and Ratnaraja T., “Modeling and performance of uplink cache-enabled massive MIMO heterogeneous networks,” IEEE Transactions on Wireless Communications, vol. 17, no. 12, pp. 8136–8149, 2018. https://doi.org/10.1109/TWC.2018.2874229

30. Sharma S. K. and Wang X., “Distributed caching enabled peak traffic reduction in ultra-dense IoT networks,” IEEE Communications Letters, vol. 22, no. 6, pp. 1252–1255, 2018. https://doi.org/10.1109/LCOMM.2018.2810211

31. Teng L., Yu X., Tang J., and Liao M., “Proactive caching strategy with content-aware weighted feature matrix learning in small cell network,” IEEE Communications Letters, vol. 23, no. 4, pp. 700–703, 2019. https://doi.org/10.1109/LCOMM.2019.2896028

32. Song F.et al., “Probabilistic caching for small-cell networks with terrestrial and aerial users,” IEEE Transactions on Vehicular Technology, vol. 68, no. 9, pp. 9162–9177, 2019. https://doi.org/10.1109/TVT.2019.2929839

33. Sufyan M. M. A.-E.et al., “Duplication elimination in cache-uplink transmission over B5G small cell network,” EURASIP Journal on Wireless Communications and Networking, vol. 2021, no. 1, pp. 1–24, 2021. https://doi.org/10.1186/s13638-021-02059-2

34. Liu J. and Sun S., “Energy efficiency analysis of cache-enabled cooperative dense small cell networks,” IET Communications, vol. 11, no. 4, pp. 477–482, 2017. https://doi.org/10.1049/iet-com.2016.0680

35. Zhao J., Wang W., Qu H., Zhao S., and Ren G., “Joint caching policies for optimizing energy costs/off-loading probability for D2D and millimeter-wave small cell underlaying cache-enabled networks,” Transactions on Emerging Telecommunications Technologies, vol. 31, no. 3, p. e3804, 2020. https://doi.org/10.1002/ett.3804

36. Liu D. and Yang C., “Caching Policy Toward Maximal Success Probability and Area Spectral Efficiency of Cache-Enabled HetNets,” in IEEE Transactions on Communications, vol. 65, no. 6, pp. 2699–2714, June 2017. https://doi.org/10.1109/TCOMM.2017.2680447

37. Hou R., Huang K., Xie H., Li H., and Li H., “Caching and resource allocation in small cell networks,” Computer Networks, p. 107100, 2020. https://doi.org/10.1016/j.comnet.2020.107100

38. C. Zhang, H. Wu, H. Lu, and J. Liu, “Throughput analysis in cache-enabled millimeter wave HetNets with access and backhaul integration,” in 2020 IEEE Wireless Communications and Networking Conference (WCNC), 2020, pp. 1-6: IEEE.
46. Liu C., Zhang H., Ji H. and Li X., “MEC-assisted flexible transcoding strategy for adaptive bitrate video streaming in small cell networks,” in China Communications, vol. 18, no. 2, pp. 200–214, Feb. 2021. https://doi.org/10.23919/JCC.2021.02.013

47. S. Mahboob, K. Kar, and J. J. a. p. a. Chakareski, “Decentralized Collaborative Video Caching in 5G Small-Cell Base Station Cellular Networks,” 2021.

48. T. Nie, J. Luo, L. Gao, F.-C. Zheng, and L. Yu, “ Cooperative Edge Caching in Small Cell Networks with Heterogeneous Channel Qualities,” in 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), 2020, pp. 1-6: IEEE.

49. Yang X. and Thomos N., “An approximate dynamic programming approach for collaborative caching,” Engineering Optimization, vol. 53, no. 6, pp. 1005–1023, 2021. https://doi.org/10.1080/0305215X.2020.1767098

50. Wu D., Liu B., Yang Q., and Wang R., “Social-aware cooperative caching mechanism in mobile social networks,” Journal of Network and Computer Applications, vol. 149, p. 102457, 2020. https://doi.org/10.1016/j.jnca.2019.102457

51. Lin P., Song Q., Song J., Jamalipour A., and Yu F. R., “Cooperative caching and transmission in CoMP-integrated cellular networks using reinforcement learning,” IEEE Transactions on Vehicular Technology, vol. 69, no. 5, pp. 5508–5520, 2020. https://doi.org/10.1109/TVT.2020.2981657

52. P. Shu and Q. Du, “Group Behavior-Based Collaborative Caching for Mobile Edge Computing,” in 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2020, vol. 1: IEEE, pp. 2441-2447.

53. H. Wu, J. Li, J. Zhi, Y. Ren, and L. Li, “Edge-oriented Collaborative Caching in Information-Centric Networking,” in 2019 IEEE Symposium on Computers and Communications (ISCC), 2019: IEEE, pp. 1-6.

54. Ren D., Gui X., Zhang K., and Wu J., “Hybrid collaborative caching in mobile edge networks: An analytical approach,” Computer Networks, vol. 158, pp. 1–16, 2019. https://doi.org/10.1016/j.comnet.2019.04.026

55. X. Yang and N. Thomos, “A rolling-horizon dynamic programming approach for collaborative caching,” arXiv preprint arXiv:1907.13516, 2019.

56. Xu X. and Tao M., “Modeling, Analysis, and Optimization of Caching in Multi-Antenna Small-Cell Networks,” IEEE Transactions on Wireless Communications, vol. 18, no. 11, pp. 5454–5469, 2019. https://doi.org/10.1109/TWC.2019.2936390

57. Zhang S., Sun W., and Liu J., “Spatially cooperative caching and optimization for heterogeneous network,” IEEE Transactions on Vehicular Technology, vol. 68, no. 11, pp. 11260–11270, 2019. https://doi.org/10.1109/TVT.2019.2941115

58. Y.-T. Wang, Y.-Z. Cai, L.-A. Chen, S.-J. Lin, and M.-H. Tsai, “Backhaul-Based Cooperative Caching in Small Cell Network,” in International Conference on Advanced Information Networking and Applications, 2019, pp. 725-736: Springer.

59. V. Bioglio, F. Gabry and I. Land, “Optimizing MDS Codes for Caching at the Edge,” 2015 IEEE Global Communications Conference (GLOBECOM), 2015, pp. 1-6.

60. Sloss A., Symes D., and Wright C., ARM system developer’s guide: designing and optimizing system software. Elsevier, 2004.

61. C. Bernardini, T. Silverston and O. Festor, “A Comparison of Caching Strategies for Content Centric Networking,” 2015 IEEE Global Communications Conference (GLOBECOM), 2015, pp. 1-6.

62. F. Tian, X. Chen and Z. Zhang, “Robust Design for Massive Access in B5G Cellular Internet of Things,” 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), Xi’an, China, 2019, pp. 1-6.

63. F. Tian and X. Chen, “Energy-Efficient Design for Massive Access in B5G Cellular Internet of Things,” 2020 IEEE Wireless Communications and Networking Conference (WCNC), 2020, pp. 1-6.

64. Qadri M. and McDonald-Maier K., “Data cache-energy and throughput models: design exploration for embedded processors,” EURASIP journal on embedded systems, vol. 2009, no. 1, p. 725438, 2009. https://doi.org/10.1051/2009725438

65. Chen L.-M., Zou X.-C., Lei J.-M., and Liu Z.-L., “Dynamic cache resources allocation for energy efficiency,” The Journal of China Universities of Posts and Telecommunications, vol. 16, no. 1, pp. 121–126, 2009. https://doi.org/10.1016/S1005-8885(08)60191-0

66. Yan Z., Chen S., Ou Y., and Liu H. J. I. A., “Energy efficiency analysis of cache-enabled two-tier HetNets under different spectrum deployment strategies,” vol. 5, pp. 6791–6800, 2017.

67. Sboui L., Rezki Z., Sultan A. and Alouini M., “A New Relation Between Energy Efficiency and Spectral Efficiency in Wireless Communications Systems,” in IEEE Wireless Communications, vol. 26, no. 3, pp. 168–174, June 2019. https://doi.org/10.1109/MWC.2019.1800161
68. He Y., Wang M., Yu J., He Q., Sun H., and Su F. J. E., “Research on the Hybrid Recommendation Method of Retail Electricity Price Package Based on Power User Characteristics and Multi-Attribute Utility in China,” Energies, vol. 13, no. 11, p. 2693, 2020. https://doi.org/10.3390/en13112693

69. Christen P., “The data matching process,” in Data matching: Springer, 2012, pp. 23–35.

70. Çelebi H., Yapıcı Y., Güvenç İ. and Schulrinne H., “Load-Based On/Off Scheduling for Energy-Efficient Delay-Tolerant 5G Networks,” in IEEE Transactions on Green Communications and Networking, vol. 3, no. 4, pp. 955–970, Dec. 2019. https://doi.org/10.1109/TGCN.2019.2931700

71. Chang K.-C., Chu K.-C., Wang H.-C., Lin Y.-C., and Pan J.-S., “Energy saving technology of 5G base station based on Internet of Things collaborative control,” IEEE Access, vol. 8, pp. 32935–32946, 2020. https://doi.org/10.1109/ACCESS.2020.2973648

72. Luo Y., Shi Z., Bu F. and Xiong J., “Joint Optimization of Area Spectral Efficiency and Energy Efficiency for Two-Tier Heterogeneous Ultra-Dense Networks,” in IEEE Access, vol. 7, pp. 12073–12086, 2019. https://doi.org/10.1109/ACCESS.2019.2891551

73. Liu J., Li D. and Xu Y., “Collaborative Online Edge Caching With Bayesian Clustering in Wireless Networks,” in IEEE Internet of Things Journal, vol. 7, no. 2, pp. 1548–1560, Feb. 2020. https://doi.org/10.1109/JIOT.2019.2956554