Research on Cache Placement Optimization and Popularity-based Content Prefetching Strategy in LEO Satellite Networks

Yaqin Xie, Chengshuai Zhou, Zhongyu Liu, Yu Zhang, Jianyue Zhu, and Junmin Wu

***Abstract*—Low Earth Orbit (LEO) satellite networks, due to their wide coverage and low latency, have shown great potential in global content distribution. However, the limited caching resources and increasing user demands present significant challenges for the optimization of content caching strategies. Named Data Networking (NDN), by decoupling content from addresses, enables more efficient use of the caching resources in satellite nodes, thereby reducing transmission latency. In this work, we first provide the optimized placement of cache nodes in LEO satellite networks and an axis-based cache placement strategy is proposed. Furthermore, a dynamic prefetching strategy based on content popularity is introduced, integrating the calculation of the Value of Popularity (VoP) matrix to optimize content selection for caching. Simulation results demonstrate that, compared to the classical LCE caching strategy, the proposed strategy significantly reduces the average path length, decreases content transmission latency by approximately 10%, and enhances the cache hit rate by approximately 12%.**

This work was supported by the the National Natural Science Foundation of China (NSFC) under Grant 62001238 and Major Science and Technology Project of Jiangsu Province under Grant BG2024002.

Yaqin Xie, Chengshuai Zhou, Zhongyu Liu, Yu Zhang and Jianyue Zhu are with the School of Electronics and Information Engineering, Nanjing University of Information Science and Technology, Nanjing, 210044, China (e-mail:xyq@nuist.edu.cn, 202212490196@nuist.edu.cn, 202312490220@nuist.edu.cn, zhangyu@nuist.edu.cn, zhujy@nuist.edu.cn,)

Junmin Wu are with Electric Power Wireless Communication Technology Laboratory, China Electric Power Research Institute, Nanjing, 210003, China (e-mail: wujunmin@epri.sgcc.com.cn)

***Index Terms*—Low earth orbit (LEO) satellite, Named data network, Content popularity, Caching strategy.**

I. Introduction

In remote areas and extreme environments, the coverage of terrestrial networks is limited, making it difficult to meet the communication needs of people in these regions. Due to their extensive coverage, lower construction, and operational costs, Low Earth Orbit (LEO) satellite networks are an ideal choice for content distribution services. Through satellite networks, users can achieve seamless connectivity anywhere. Utilizing satellite-terrestrial integrated networks [1] to provide global content distribution services is a new trend in the development of next-generation communication networks. However, satellite networks face several challenges during transmission, such as high transmission latency, high error rates, limited storage and processing capabilities, and uneven network traffic loads. Named Data Networking (NDN), through its unique in-network caching technology, can effectively address these issues when applied to satellite networks, enabling diverse and efficient content distribution. The satellite networks using NDN will not only reduce content transmission latency but also improve the utilization of network resources [2].

When users within the same region have similar content requests, repeated backhaul links can cause imbalanced loads,

leading to network congestion and reducing the quality of user experience [3], [4]. Fortunately, cache-enabled networks can reduce content transmission latency and minimize wasted backhaul bandwidth. Additionally, different types of content exhibit varying request behaviors [5], [6], such as video streaming, web content, and remote sensing data. These different request behaviors impact overall content caching, as content may compete for scarce caching resources, resulting in inefficiencies and longer user response delays. Caching technology has proven to be an effective method for addressing these issues. However, the existing caching strategies in satellite networks still face multiple limitations. Traditional caching mechanisms, such as Leave Copy Everywhere (LCE), Least Recently Used (LRU), and First-In-First-Out (FIFO), often do not consider the highly dynamic topology of LEO satellite networks, leading to inefficient cache utilization and frequent cache invalidation. As satellites move rapidly across different coverage areas, cached content may become inaccessible to users, reducing cache hit rates and increasing retrieval delays. Furthermore, most existing strategies primarily rely on static content request frequencies to determine caching priorities, neglecting the fact that content popularity evolves dynamically over time due to changes in user behavior and network conditions. This lack of adaptability often results in suboptimal caching decisions, where outdated or low-demand content occupies valuable cache space, while emerging popular content is not proactively stored at optimal locations. Additionally, the absence of a structured cache placement strategy means that content is often scattered randomly across satellite nodes, leading to redundant data storage and inefficient routing paths, which in turn increase network congestion and backhaul dependency.

The Information-Centric Networking (ICN) is a promising new network architecture. NDN is a widely researched pull-based ICN architecture, and its content-centric nature has garnered significant attention for applications that require content retrieval, especially in LEO satellite networks. NDN abandons the traditional layered network architecture, such as the application layer and network layer. Instead, NDN adopts a content-centric protocol stack, placing content at the core of network design, where content naming becomes the sole identifier for data transmission across the network. With ongoing improvements in satellite hardware and the reduction in satellite equipment costs, content caching and computing onboard satellites have become routine operations within satellite systems. By decoupling content from addresses, NDN deploys reusable content closer to users at satellite nodes, thereby reducing the transmission distance and avoiding redundant transmissions of the same content [7]. Consequently, the introduction of NDN in LEO satellite networks can effectively leverage its built-in caching mechanisms to reduce data transmission latency and enhance the utilization of network resources. By deploying cache nodes within the satellite network, NDN can store popular content closer to users, thereby reducing the number of hops and transmission delays for data requests. This approach not only optimizes network performance but also alleviates the load on inter-satellite links, improving the overall quality of service of the network.

Caching technology in satellite networks faces two primary challenges: how to select cache locations and how to choose appropriate content for caching. To address these challenges, this paper focuses on the network cache deployment strategy in LEO satellite networks under the NDN architecture and proposes a hybrid caching strategy based on axis placement and popularity-aware content. The strategy calculates optimal cache locations using a fast algorithm and combines the calculation of content popularity values through the Value of Popularity (VoP) matrix to cache high-popularity content and prefetch content at cache nodes, thereby improving network performance. The main contributions of this paper are as

follows:

- The proposal set out herein is for a structured and adaptive cache placement strategy with a view to optimising the layout of cache nodes in LEO satellite networks. In contrast to conventional random or uniform cache deployment strategies, our AXES (Axis-Based Placement) approach involves the selection of cache nodes along specific grid axes in a staggered layout. The efficacy of this system is predicated on the ability to route traffic through deterministic paths that invariably intersect cache nodes. This, in turn, serves to minimize average retrieval latency and balance the traffic load across the network. In comparison with the naive placement strategy, this structured approach has been shown to significantly mitigate the performance degradation that is caused by the high mobility of LEO satellites.
- The placement optimisation algorithm was designed to be both fast and effective. It dynamically refines the positions of cache nodes based on the principle of minimising the total service distance in the grid. By iteratively evaluating Manhattan-distance-based cost functions, the algorithm adaptively converges to a near-optimal node configuration. It has been demonstrated that the computational complexity remains linear with the number of candidate positions, thus rendering it suitable for large-scale LEO constellations.
- The present study proposes the introduction of a dynamic VoP matrix, with the objective of assessing the real-time popularity of content. In contrast to conventional popularity metrics that are predicated exclusively on request frequency, the present model incorporates request intervals and observation delays, whilst utilizing an exponential decay function with exponential weighted moving average (EWMA) smoothing. This enables the capture of temporal variation and the aging of content interest, thus facilitating more accurate and responsive caching and prefetching decisions. The simulation results demonstrate that this dynamic VoP-based approach significantly improves both the cache hit ratio and route efficiency compared to static popularity-based strategies.

The remainder of this paper is organized as follows. Section II discusses related work. A detailed description of the system model is provided in Section III. Section IV focuses on the proposed caching optimization strategy. Section V presents the simulation results. Finally, Section VI concludes the paper.

II. Related Work

In this section, the content caching strategies in NDN and the mechanisms for calculating content popularity are primarily introduced.

A. Content Caching Strategies

Currently, leveraging the caching and computing capabilities of satellite nodes to enhance the overall transmission capacity of networks has attracted attention from the academic community. D'oro et al. [8] introduced a novel information-centric caching strategy for satellite networks, leveraging the inherent broadcasting nature of satellite communications in conjunction with user preferences, to enhance the availability of communication resources. To meet the internet access demands of users in remote areas, Zhong et al. [9] designed a social distance-based helper designation model and derived the optimal video streaming cache layout using an approximation algorithm. Yang et al. [10] introduced a satellite-terrestrial collaborative caching network architecture based on mobile edge computing, which reduces content delivery latency through multi-tier node content sharing and hierarchical coordination of cached content. Zhu et al. [11] studied a multi-tier collaborative caching framework for satellite-terrestrial integrated networks and designed both non-cooperative and cooperative caching

strategies to minimize content retrieval latency. Liu et al. [12] proposed a LEO satellite network caching placement method based on regional user interest awareness, which divides the network's caching space by analyzing content similarity and user preferences, aiming to minimize response latency and achieve optimal content distribution.

In summary, caching technologies in NDN have made significant progress in satellite network applications. However, as the scale of LEO satellite networks expands and the topology becomes increasingly dynamic, existing caching strategies still face many challenges. Therefore, this paper proposes an axis-based caching placement strategy that optimizes the layout of cache nodes to achieve more efficient content distribution in complex network environments, significantly reducing transmission latency.

B. Popularity-based Mechanism

Popularity is widely used in content caching within networks, as caching popular content can improve cache hit rates, reduce content retrieval latency, and decrease network resource consumption. By leveraging popularity, the system can prioritize caching highly popular content at nodes closer to the users. When users, acting as consumers, make requests, the exchange of interest and data packets between consumer nodes is correspondingly reduced. In existing research, several content popularity calculation mechanisms have been proposed [13]. These mechanisms utilize the content popularity to prefetch content to nodes closer to consumers before the consumer requests the content.

In [14], a diversity-aware content caching mechanism is introduced, which takes into account content popularity, content lifecycle, and cache availability when making caching decisions. This mechanism introduces a content popularity index, defining popularity as the ratio of the number of interest requests for a specific content to the total number of interest requests for all content. In [15], a popularity-based autonomous caching mechanism is described, where the content with the highest popularity value, referred to as the most popular content, is cached. In this mechanism, each node locally stores pairs of "content name" and "content popularity" information. Content popularity is defined as the ratio of the number of requests received by a node for specific content to the total number of content requests generated in the network. Caching decisions are made based on the popularity table, and content is cached on nodes once its popularity value exceeds a set threshold. To further expand the concept of content popularity for more efficient decision-making, [16] proposes a dynamic fine-grained caching mechanism. This mechanism dynamically adjusts the popularity threshold based on the frequency of interest requests and the availability of cache space. Content caching decisions are governed by this popularity threshold— the larger the cache space, the lower the threshold, and vice versa. In [17], the relationship between content freshness and service latency is explored, where cache updates and content delivery are decoupled by bandwidth splitting. This scheme updates cached content in a cyclical manner, maintaining content freshness when there is a certain probability of a request from the user. In [18], a distributed content popularity-based caching mechanism is introduced, wherein popular content is cached at neighboring nodes, distributing network traffic among communication nodes and preventing congestion at a single content provider. In [19], Zhang design a name-based dynamic information forwarding strategy for Named Data Networking (NDN) in the Internet of Vehicles (IoV), which integrates network topology prediction and name-based routing to reduce latency and improve delivery stability—demonstrating strong adaptability in highly dynamic networks. In [20], Zhang propose a content-aware proportional caching strategy (CA-prop), which allocates limited LEO satellite cache resources based on content access fre-

quency and importance. Their approach significantly enhances cache hit rates and transmission efficiency in dynamic satellite network environments. While these existing popularity calculation mechanisms have contributed substantially to improving network caching performance, most of them rely primarily on static indicators—such as cumulative request frequency or average access count—without adequately accounting for temporal variation in user behavior. This often limits their responsiveness in dynamic environments. To address this, recent studies have explored incorporating time-aware metrics. For example, Zhang et al. [21] employed an exponential decay function to account for popularity freshness in vehicular networks, while Liu et al. [22] used temporal intervals to adjust edge cache priorities. However, these models typically lack smoothing strategies or fail to integrate multiple temporal dimensions such as observation delay, which are critical in dynamic topologies like LEO satellite networks.

Based on the aforementioned issues, this paper proposes an adaptive popularity calculation mechanism capable of capturing the time-varying nature of content popularity. By utilizing the concept of content popularity, this mechanism can not only dynamically adjust cached content but also prefetch popular content closer to the users in advance. As a result, it effectively reduces content transmission latency and improves the efficiency of network resource utilization.

III. System Model

The system archeitecture of the proposed scheme in this paper is given in Fig.1. Its service content originates from a ground-based content source server, and the content is transmitted to user terminals via the LEO satellite network. Specifically, this LEO satellite network model comprises a content source server, a LEO satellite constellation, multiple user terminals, and several ground stations (acting as gateways) that connect users, the content source server, and the

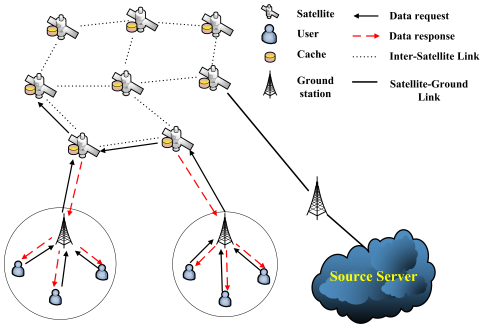


Fig. 1: System architecture diagram.

satellites. Satellite nodes communicate with each other through Inter-Satellite Links (ISLs) and communicate with ground stations via Satellite-Ground Links (SGLs). Typically, satellites in a LEO constellation are distributed across multiple orbital planes, all of which share the same inclination. Satellites within the same orbital plane maintain relatively stable positions, enabling continuous connections with satellites ahead and behind them in the same orbit. Additionally, the distances between orbital planes remain stable, allowing each satellite to establish communication links with satellites in adjacent orbits. Under this configuration, each satellite is equipped with four ISLs: two connecting neighboring satellites in the same orbit, forming horizontal links along the orbit, and two connecting satellites in adjacent orbital planes, creating vertical cross-orbital links. This configuration results in a grid-like topology for the satellite network, with connections both within and between orbital planes.

In the NDN-based satellite network, each satellite node supports content caching, enabling it to store and forward content copies. When a user sends a request, it is first directed to the connected satellite node. If the requested content is cached locally, the node responds directly; otherwise, the request is forwarded via ISLs to other satellites. If no satellite holds the content, the request is relayed to the ground server via SGL. Upon receiving the content, the data is returned along the same path and cached at appropriate intermediate nodes.

To simplify the constructed LEO satellite network model,

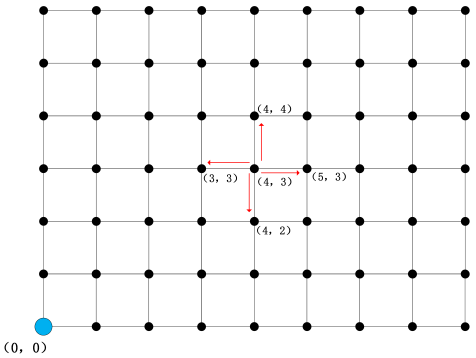


Fig. 2: Simplified satellite topology.

a simplified topology is obtained, as shown in Fig 2. In this figure, black nodes represent ground stations or satellite nodes connected to ground stations, responsible for forwarding content requests to other nodes within the satellite network. Blue nodes represent the original content producers, which are the sources of content generation. Except for the edge nodes, the remaining satellite nodes are connected to neighboring satellite nodes through four ISLs. When a user requests content, if the requested content is not found in other cache nodes, it will ultimately be provided and responded to by the blue node.

Let the two-dimensional coordinates of the n-th satellite node be denoted as (x_n, y_n), and let N be the total number of satellites. The model illustrated in the diagram can be represented by an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$ is the set of satellites. Here, n n ranges from 1 to N, representing each satellite node in the network. $\mathcal{E}$ is the set of edges representing the Inter-Satellite Links (ISLs) between these nodes. In this model, $\mathcal{U} = \{u_1, u_2, \cdots u_M\}$ denotes the set of user terminals, with M indicating the total number of users. The set of content files is denoted by $\mathcal{F} = \{f_1, f_2, \cdots f_R\}$, where R represents the total number of content items. The distance between any two satellite nodes is measured using the Manhattan distance. This distance metric is appropriate as it accounts for the relative movement of satellite nodes along the horizontal and vertical directions within the orbital plane and between different orbital planes, making it suitable for the grid-like topology of LEO satellite networks. This distance measurement method ensures more rational and efficient path planning within the grid network. Thus, the distance between the n-th satellite node and the n'-th satellite node can be expressed as:

$$d(n, n') = |x_n - x_{n'}| + |y_n - y_{n'}| \tag{1}$$

where n and $n' \in [1, N]$. When the m-th user requests the r-th content chunk, this is denoted as $req(u_m, f_r)$. Here, req indicates a request operation, u_m is the user making the request, and f_r is the specific content chunk being requested. If a satellite node along the data request path has cached the r-th content chunk, that node can directly respond to the user's request. In LEO satellite networks, due to the limited and widely distributed storage resources of the nodes, designing an efficient caching model is crucial for optimizing content delivery. The caching strategy proposed in this paper assumes that all satellite nodes in the network have the option to either enable or disable caching functionality. This assumption provides a basis for the flexible configuration of caching resources. Let a Boolean variable $C(n, f_r)$ denote whether satellite node n caches content f_r. If $C(n, f_r) = 1$, it indicates that the node is caching the content. Since each node's cache space is limited, it is essential to ensure that the total amount of content cached by a node does not exceed its maximum capacity. Assuming the size of each content is denoted by ω, and β_n represents the maximum cache capacity of node n, the following constraint must be satisfied for $n \in \{1, 2, \ldots N\}$:

$$\sum_{r=1}^{R} C(n, f_r)\omega \leq \beta_n \tag{2}$$

IV. Caching Placement and Content Prefetching Strategy Design

A. Axes-based Cache Placement

To achieve efficient cache node deployment, this paper adopts an axes-based cache placement strategy. The core idea of this strategy is to consider only the nodes located on the axes in the grid network as cache nodes, thereby simplifying the routing process and improving caching efficiency. Under this strategy, interest packets are first forwarded directly to the nearest axis and then transmitted along the axis to the producer node. This approach ensures that interest packets will encounter a cache node during transmission, reducing data transmission delay.

Specifically, when a node knows the exact location of the nearest cache node or the axis it resides on, this information can be utilized to determine the optimal forwarding path for interest packets. Even forwarding an interest packet to a more distant axis may be worthwhile if it leads to a closer cache node. In this way, the selection of cache nodes and routing decisions are simplified, ensuring that interest packets are transmitted along the shortest path, thereby improving the overall performance of the network.

As demonstrated in [23], placing caches in a staggered manner minimizes the total data transmission path length. Therefore, the cache placement in the following section also adopts a staggered configuration. Fig.3 illustrates the placement of cache nodes and their corresponding service areas. The yellow nodes (i.e., c_1, c_2, c_3, and c_4 represent the selected cache nodes, with the red boxes indicating the service areas of these cache nodes. The solid blue circles represent producer nodes, which are located at the center of the network. It can be observed that cache nodes c_1 and c_2 are close to the central area of the network and have smaller service areas, while cache nodes c_3 and c_4 are located near the network's edge and

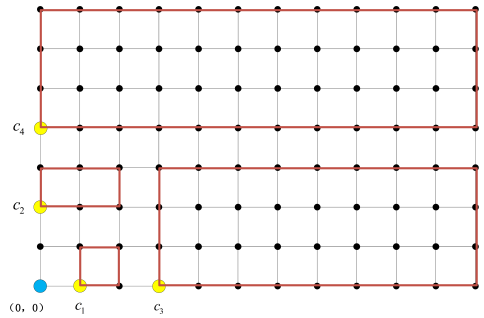


Fig. 3: Cache node placement and corresponding service areas.

have relatively larger service areas. Therefore, to minimize the average distance between other satellite nodes in the network and the cache nodes, the positions of the cache nodes must be optimized.

To quantify the size of the service area of a cache node, we define two ordered sets of cache nodes along the axes. One along the vertical axis $\mathcal{X} = \left\{\left(x_1^{hor}, 0\right), \cdots, \left(x_H^{hor}, 0\right)\right\}$ and the corresponding set along the horizontal axis $\mathcal{Y} = \{(0, y_1^{vert}), \cdots, (0, y_V^{vert})\}$, where H and V represent the number of cache nodes along the horizontal and vertical axes, respectively. Let N denote the total number of cache nodes in the entire network, where $Q = H + V$. Then, the set of satellite nodes served by cache node $(x_h^{hor}, 0)$ is defined as $\mathcal{A}\left(x_h^{hor}\right)$, such that:

$$\mathcal{A}\left(x_h^{hor}\right) = \left\{(x, y) \,|x_h^{hor} \leq x < x_{h+1}^{hor}, y < y_k^{vert}\right\} \tag{3}$$

where $y_k^{vert} = \max(y_j^{vert} | (0, y_j^{vert}) \in \mathcal{Y} \text{ and } y_j^{vert} < x_{h+1}^{hor})$. Similarly, let $\mathcal{A}\left(y_v^{vert}\right)$ represent the set of nodes served by cache node $(0, y_v^{vert})$. Then, $\mathcal{A}\left(y_v^{vert}\right)$ can be expressed as:

$$\mathcal{A}\left(y_v^{vert}\right) = \left\{(x, y) \,|y_v^{vert} \leq y < y_{v+1}^{vert}, x < x_l^{hor}\right\} \tag{4}$$

where $x_l^{hor} = \max(x_i^{hor} | (0, x_i^{hor}) \in \mathcal{X} \text{ and } x_i^{hor} < y_{v+1}^{vert})$. From (3) and (4), it is evident that the service area of each cache node can be determined by the following three locations: the cache node itself, the position of the next cache node on the same axis, and the position of a single cache node on the other

axis. For example, in Fig.4, the service area of cache node c_2 is determined by c_2, c_4 and c_3. Next, we define $\left\|\mathcal{A}\left(x_h^{hor}\right)\right\|$ as the sum of the distances from all satellite nodes in the service area of cache node $(x_h^{hor}, 0)$ to that cache node. Thus, we can get:

$$\begin{aligned}\left\|\mathcal{A}\left(x_h^{\text{hor}}\right)\right\| &= \left\|\left(x_h^{\text{hor}}, x_{h+1}^{\text{hor}}, y_k^{\text{vert}}\right)\right\| \\ &= \sum_{n=x_h^{\text{hor}}}^{x_{h+1}^{\text{hor}}-1} \sum_{n'=0}^{y_k^{\text{vert}}-1} d\left((n,n'),\left(x_h^{\text{hor}},0\right)\right) \\ &= \sum_{n=x_h^{\text{hor}}}^{x_{h+1}^{\text{hor}}-1} \sum_{n'=0}^{y_k^{\text{vert}}-1} \left(n+n'-x_h^{\text{hor}}\right) \\ &= y_k^{\text{vert}}\left(x_{h+1}^{\text{hor}}-x_h^{\text{hor}}\right)\frac{x_{h+1}^{\text{hor}}-x_h^{\text{hor}}+y_k^{\text{vert}}-2}{2}\end{aligned} \tag{5}$$

Similarly, $\|\mathcal{A}\left(y_v^{vert}\right)\|$ can also be expressed as:

$$\begin{aligned}\left\|\mathcal{A}\left(y_v^{\text{vert}}\right)\right\| &= \left\|\left(y_v^{\text{vert}}, y_{v+1}^{\text{vert}}, x_l^{\text{hor}}\right)\right\| \\ &= x_l^{\text{hor}}\left(y_{v+1}^{\text{vert}}-y_v^{\text{vert}}\right)\frac{y_{v+1}^{\text{vert}}-y_v^{\text{vert}}+x_l^{\text{hor}}-2}{2}\end{aligned} \tag{6}$$

Let the set of cache nodes be denoted as $\mathcal{C} = \{c_1, c_2, \cdots c_q, \cdots c_Q\}$, where c_q represents the position of the q-th cache node. Under the staggered placement strategy, the layout of cache nodes follows a combinatorial arrangement, ensuring that the service area of each cache node covers nodes along both the horizontal and vertical axes. Therefore, we define the total transmission path length as $D(\mathcal{C})$, which can be expressed as the sum of the combinatorial distances between multiple pairs of cache nodes. Specifically, this combination includes two adjacent cache nodes on the same axis and one cache node on the other axis, as shown in Eq.(7):

$$D(\mathcal{C}) = \|c_1, c_3, c_2\| + \|c_2, c_4, c_3\| + \|c_3, c_5, c_4\| + \cdots \tag{7}$$

To further specify the calculation of the total transmission path length, we can decompose these combinatorial distances into the sum of transmission distances along the horizontal and vertical axes. By integrating the definition of service areas and the staggered arrangement of cache nodes, we can aggregate the distances calculated for the service areas along both the horizontal and vertical axes. This leads to the expression for the total transmission path length, as shown in Eq.(8).

$$D(\mathcal{C}) = \sum_{h=1}^{H}\left\|\mathcal{A}\left(x_h^{hor}\right)\right\| + \sum_{v=1}^{V}\left\|\mathcal{A}\left(y_v^{vert}\right)\right\| \tag{8}$$

Therefore, the average path length from all nodes in the network to the cache nodes, denoted as D_{avg}, can be expressed as:

$$D_{avg} = \frac{D(\mathcal{C})}{Q} \tag{9}$$

B. Optimized Placement Algorithm

To further optimize the placement of cache nodes, this paper introduces a fast algorithm designed to determine the optimal positions of Q cache nodes along the coordinate axes to minimize the average path length from other satellite nodes to the cache nodes. This optimization problem is equivalent to finding the minimum value of D_{avg} in equation 9. The core idea of the algorithm is to iteratively adjust the positions of the cache nodes to find a configuration that reduces the total transmission path length for the network.Specifically, the algorithm begins by initializing the positions of the cache nodes and calculating the total path cost for the current configuration. Then, the algorithm enters a loop where, in each iteration, it sequentially checks each cache node and attempts to move its position by one unit along the axis. If the new position results in a reduction in the total transmission cost, the algorithm updates the cache node's position and the minimum cost marker. If no further improvements are possible, the algorithm exits the loop and concludes the optimization process. The detailed steps of the algorithm are as follows:

The application of Algorithm 1 is now demonstrated in a specific network environment. Taking the 12×8 grid network from Fig.3 as an example, the cache nodes were initially placed uniformly along the axis close to the content producers, with initial coordinates of $c_1(1,0)$, $c_2(0,2)$, $c_3(3,0)$, and

Algorithm 1 Optimal cache placement

Input: Initial placement of cache nodes $\mathcal{C} = \{c_1, c_2, \cdots, c_Q\}$; $lowest\_cost \leftarrow D(\mathcal{C})$

Output: Optimal cache node placement combination for minimum cost **while** *true* **do**

end

improved $\leftarrow$ false **for** A **do**

l

end

l$q \in \{Q, \ldots 1\}$ **while** $c_q + 1 < c_{q+1}$ **do**

end

$\mathcal{C}' = \{c_1, c_2, \cdots c_q + 1, c_{q+1}, \cdots c_Q\}$

$current\_cost \leftarrow \mathrm{D}(C')$ **if** $current\_cost < lowest\_cost$ **then**

end

$lowest\_cost \leftarrow current\_cost$

$\mathcal{C} \leftarrow \mathcal{C}'$

improved $\leftarrow$ true **else**

end

break while

=0

$c_4(0, 4)$. At this stage, the average path length from all other nodes in the service areas of each cache node to the cache node itself was calculated using (5) and (6) to be

$$\begin{aligned} D_{avg}^{initial} &= \frac{\|1,3,2\|+\|2,4,3\|+\|3,12,4\|+\|4,8,12\|}{4} \\ &= \frac{4+9+198+336}{4} \\ &\approx 137 \end{aligned} \tag{10}$$

Based on this initial layout, Fig.4 illustrates the deployment of cache node positions after adjustment using Algorithm 1. After parallel computation and multiple configuration attempts, the positions of the cache nodes were updated to $c_1(0, 1)$, $c_2(4, 0)$, $c_3(0, 5)$, and $c_4(9, 0)$. Comparing Fig.3 and Fig. 4, it can be observed that the optimized distribution of cache nodes is more uniform and covers a larger area. The average path length from nodes in the network to cache nodes has been reduced to

$$\begin{aligned} D_{avg}^{optimal} &= \frac{\|1,5,4\|+\|4,9,5\|+\|5,8,9\|+\|9,12,8\|}{4} \\ &= \frac{48+100+135+108}{4} \\ &\approx 98 \end{aligned} \tag{11}$$

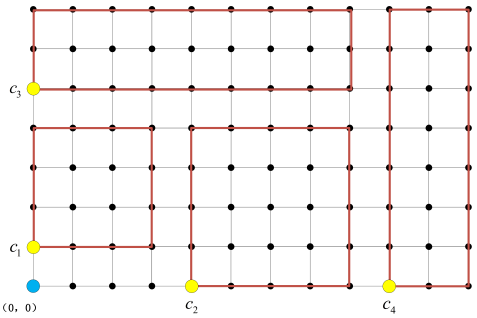


Fig. 4: Optimization of cache node placement in grid networks.

After determining the optimal cache node positions, we will further consider which specific contents to cache at these locations. Traditional caching strategies often focus solely on the number of content requests, which may be insufficient in some cases, as the temporal distribution of requests also significantly impacts content popularity. For instance, a particular sensor's data may experience high request volumes during the day but be accessed less frequently at night. Considering only request counts might result in a caching strategy that fails to dynamically adapt to such time-variant demands.

C. Content Prefetching Mechanism based on VoP

To more accurately evaluate content popularity, this section introduces a dynamic content popularity calculation mechanism and further discusses content selection for prefetching based on the VoP.

1) The Calculation of VoP: As previously mentioned, traditional static content popularity methods rely solely on the number of content requests to generate a fixed popularity value, which proves to be limited in the dynamic environment of LEO satellite networks. In LEO satellite networks, due to the high mobility of nodes and frequent topology changes, the calculation of the VoP matrix needs to comprehensively consider the time intervals and frequency of content requests to ensure that the caching strategy can quickly adapt to network state changes. The proposed dynamic VoP function considers the following three key factors: 1) Content Request Delay: The

time interval between two requests for the same content. The larger this value, the lower the frequency of content being re-requested. 2) Observation Delay: The time interval between the last content request and the current observation time. A larger value indicates that the content request is more outdated. 3) Request Count: The total number of requests for a specific content. The relationship between these factors and the VoP can be expressed as follows

$$\begin{cases} VoP\left(u_i, f_r, i\right) \propto 1/\mathcal{Y}_{i,f_r} \\ VoP\left(u_i, f_r, i\right) \propto 1/\mathcal{Z}_{i,f_r} \\ VoP\left(u_i, f_r, i\right) \propto \mathcal{I}_{u_i,f_r} \end{cases} \tag{12}$$

where $\mathcal{Y}_{i,f_r} = t_{i,f_r} - t_{i-1,f_r}$ represents the time elapsed between the i-th and $i-1$-th requests for content f_r generated by user u_i. $\mathcal{Z}_{i,f_r} = t'_{i,f_r} - t_{i,f_r}$ is the time elapsed between the last request for content f_r and the observation time t'_{i,f_r}. $\mathcal{I}_{u_i,f_r}$ is the request count for content f_r generated by user u_i. The dynamic VoP function can be calculated using the formula shown in Eq(13).

$$VoP\left(u_i, f_r, i\right) = \left(\lambda \times f\left(\mathcal{Y}_{i,f_r}\right) + \theta \times f\left(\mathcal{Z}_{i,f_r}\right)\right) \times \frac{1}{2} \tag{13}$$

where λ and θ are proportional constants, which can be adjusted to balance the influence of content request delay $\mathcal{Y}_{i,f_r}$ and observation delay $\mathcal{Z}_{i,f_r}$ on the VoP. To define the behavior of the VoP, a monotonically decreasing value update function is introduced, called the exponential decay function, as shown in Eq.(14).

$$f\left(t\right) = e^{-\frac{t}{\mathcal{I}}} \tag{14}$$

Finally, by substituting (14) into (13), we obtain:

$$VoP\left(u_i, f_r, i\right) = \left(\lambda e^{-\left(\frac{\mathcal{Y}_{i,f_r}}{\mathcal{I}_{u_i,f_r}}\right)} + \theta e^{-\left(\frac{\mathcal{Z}_{i,f_r}}{\mathcal{I}_{u_i,f_r}}\right)}\right) \times \frac{1}{2} \tag{15}$$

Although Eq.(15) provides a dynamic method for calculating content popularity, in LEO satellite networks, the highly dynamic nature of the network topology and user behavior causes content popularity to change rapidly and unpredictably. To further improve the accuracy and responsiveness of popularity calculations, this paper introduces an enhanced method based on the Exponential Weighted Moving Average (EWMA) algorithm. The improved VoP calculation formula is as follows:

$$\begin{aligned} VoP_{EW,t}(u_i, f_r, i) = (1-\gamma) \cdot VoP_{EW,t-1}(u_i, f_r, i) \\ + \gamma \cdot VoP_t(u_i, f_r, i) \end{aligned} \tag{16}$$

where γ is the smoothing coefficient, ranging between 0 and 1. It allows for significant smoothing of VoP value fluctuations while providing a more responsive reaction to sudden changes in content requests. $VoP_t(u_i, f_r, i)$ represents the VoP value at time t, and $VoP_{EW,t}(u_i, f_r, i)$ represents the VoP value at time t after applying the EWMA calculation. Fig.5 illustrates a comparison between the traditional static content popularity calculation mechanism (Fig.5(a)) and the dynamic popularity calculation mechanism (Fig.5(b)). In Fig.5(a), the user generates the first and second interests for Content1 at times t_0^1 and t_1^1, respectively, and generates the first interest for Content2 at time t_0^2. This static content popularity calculation method leads to the popularity of Content1 and Content2 remaining unchanged between t_0^2 and t_1^1, despite the differing ages of the content requests. As a result, this mechanism becomes overly rigid. Fig.5(b) demonstrates the dynamic popularity calculation method. When the user requests Interest1 for Content1 at time t_0^1, the VoP value of Content1 peaks, but this value monotonically decreases over time. However, when Interest1 for Content1 is requested again at time t_1^1, its VoP value peaks once more. The same applies to Interest2 for Content2. When the user requests Interest2 for Content2 at time t_0^2, as the VoP value of Content1 decreases over time, the VoP value of Content2 eventually surpasses that of Content1.

2) Prefetch Content Selection: After determining the most popular content using the VoP function, the next step is to consider how to proactively push this popular content to cache nodes. In the system model proposed in this paper, the network

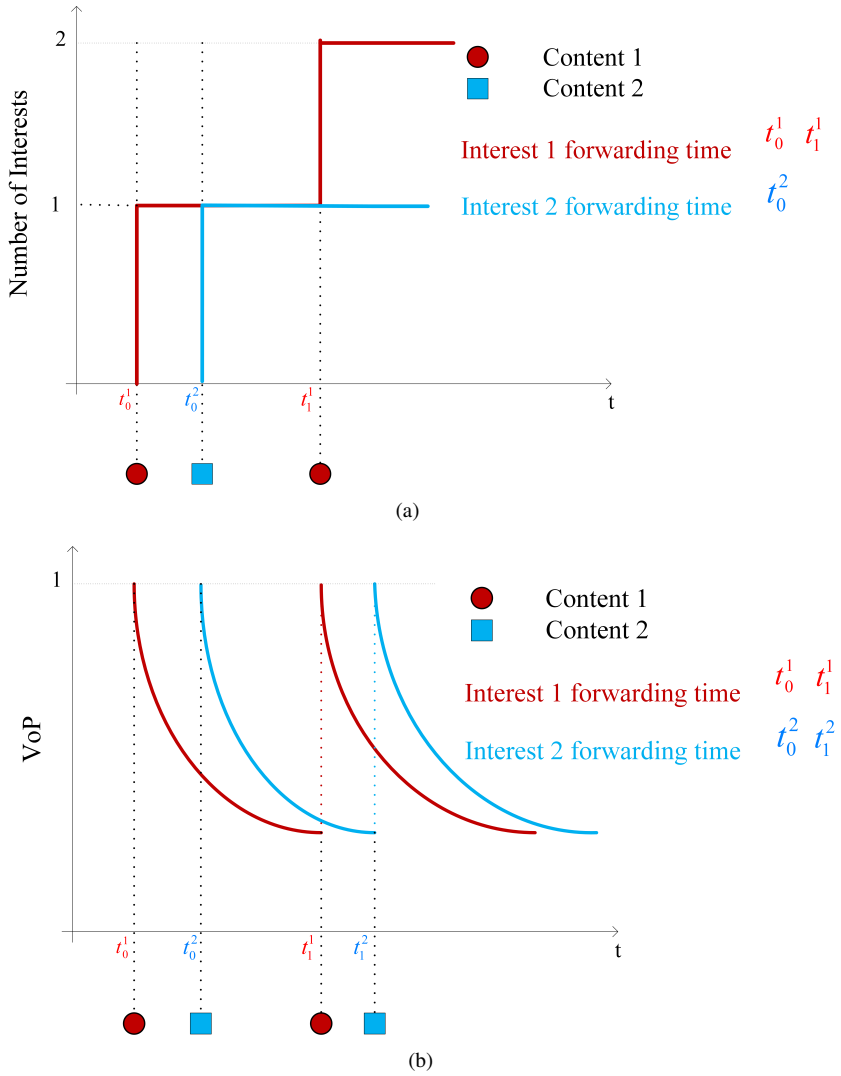


Fig. 5: Scenario with different AP distributions. (a) Traditional static popularity. (b) Dynamic popularity.

traffic is continuously monitored, maintaining a VoP vector for M users and R content items, which has been smoothed and corrected through the EWMA algorithm. This VoP vector is represented as a matrix, as shown in Eq.(17).

$$VoP_{EW}(u_i, f_r) = \begin{pmatrix} VoP(u_1,f_1) & VoP(u_1,f_2) & \cdots & VoP(u_1,f_R) \\ VoP(u_2,f_1) & VoP(u_2,f_2) & \vdots & VoP(u_2,f_R) \\ \vdots & \vdots & \ddots & \vdots \\ VoP(u_M,f_1) & VoP(u_M,f_2) & \cdots & VoP(u_M,f_R) \end{pmatrix} \quad (17)$$

Assume that different users are interested in similar content; therefore, the content request patterns of users with similar interests will be explored. To proactively prefetch content (denoted as f_l, where $f_l \in \mathcal{F}$), assume the current user u_d requests content f_r. The similarity in content request patterns is calculated by comparing the Pearson correlation coefficient between the content request patterns of the current user and other users $u_g (u_g \in \mathcal{U})$, as shown in Eq.(18).

$$S(u_g, u_d) = \frac{\sum_{n=1}^{R} \left(VoP(u_g,f_n) - \overline{VoP(u_g)}\right) \times \left(VoP(u_d,f_n) - \overline{VoP(u_d)}\right)}{\sqrt{\sum_{n=1}^{R} \left(VoP(u_g,f_n) - \overline{VoP(u_g)}\right)^2} \times \sqrt{\sum_{n=1}^{R} \left(VoP(u_d,f_n) - \overline{VoP(u_d)}\right)^2}} \quad (18)$$

where $S(u_g, u_d) \in [-1, 1]$.

The user whose content request behavior is most similar to the current user u_d, i.e., the user $u_e (u_e \in \mathcal{U})$ with the highest Pearson correlation coefficient, can be determined by

$$u_e = \arg\max_{u_g \in \mathcal{U}} \left(S_{u_g, u_d}\right) \quad (19)$$

Eq.(19) identifies the user exhibiting the highest degree of

similarity in content request patterns compared to the current user, based on the computed correlation values.

Finally, content f_l for prefetching is selected according to

$$\begin{aligned} f_l &= \underset{f_j, f_r \in \mathcal{F}}{\arg min} \left(\| VoP(u_e, f_j) - VoP(u_e, f_r) \|^2 \right) \\ s.t. \quad & VoP(u_e, f_j) > VoP(u_e, f_r) \end{aligned} \tag{20}$$

Algorithm 2 describes the process of selecting content for prefetching. Considering that user interest request patterns are similar to those shown in Fig.5(b), if the requested content is Content1, the recommended content for prefetching would be Content2.

Algorithm 2 Prefetching content selection

The use u_d requests content f_r from the network. **for** $u_g = 1 : \mathcal{U}$ **do**
    end
    $f_r = 1 : \mathcal{F}$
Update $VoP(u_d, f_r)$

for $u_g = 1 : \mathcal{U}$ **do**
    end
    Calculate the content request similarity $S(u_g, u_d)$

Find the user u_e with the highest request similarity to u_d
Prefetch content f_l, whose VoP value is closest to that of f_r.
=0

V. Evaluation and Analysis

A. Simulation Setup

To validate the effectiveness of the proposed cache optimization strategy, experiments were conducted using the ndnSIM simulation tool. It is an NS3-based [24] simulator that implements the NDN protocol stack on network nodes, enabling the deployment of NDN network functionalities and suitable for various homogeneous or heterogeneous network scenarios.

In the simulation, a 60 × 42 grid network was constructed, with one node deployed at (0,0) serving as the content provider, and satellite nodes placed at the remaining grid points. Additionally, 1000 users were randomly deployed across the network. The cache capacity of the nodes was set between 20 % and 55% of the total content, simulating varying levels of cache resource availability. User content request patterns followed a Zipf distribution, with the distribution parameter α set between 0.7 and 1.5 to reflect the concentration of user requests. The selected range of α is widely adopted in related literature to represent various content popularity distributions observed in real-world systems. Specifically, lower α values (e.g., 0.7) simulate relatively uniform content demand, while higher values (e.g., 1.5) reflect strong popularity skew where a small number of items dominate requests. This range captures realistic access patterns found in web services, content delivery networks, and mobile edge scenarios. Although the simulation is not based on a specific dataset, the chosen value of α is generally recognized as a reasonable approximation of user behavior across diverse content-centric environments. To evaluate the performance of the AXES cache placement strategy, the impact of the number of cache nodes, client quantity, and simulation duration on the average path length was analyzed. Subsequently, a control variable method was used to observe the effects of content request popularity and node cache capacity on content distribution performance. For objective assessment of different strategies and the impact of various parameters on content distribution performance, metrics such as average routing hops and cache hit rate were used. The comparison strategies include the classical cache placement strategy LCE (Leave Copy Everywhere), with cache replacement strategies of LRU (Least Recently Used) and FIFO (First In, First Out). The content prefetching strategy VoP was compared with FIFO and LRU, where both FIFO and LRU used the LCE cache placement strategy. By comparing AXES+VoP, LCE+LRU, and LCE+FIFO, the performance of the proposed cache optimization strategy was validated.

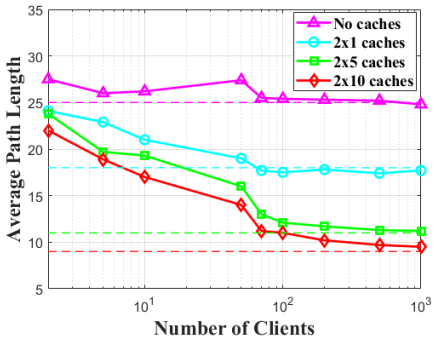


Fig. 6: Plot of average path length vs. the number of clients.

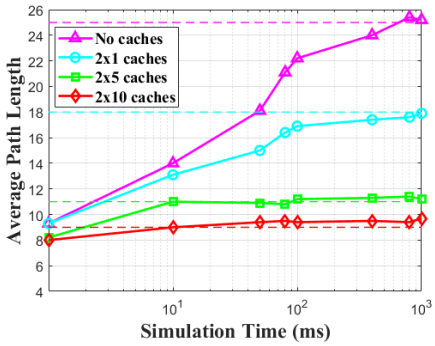


Fig. 7: Plot of average path length vs. simulation time.

B. Average Path Length

The Average Path Length (APL) refers to the average distance a client request travels to reach the nearest cache node, regardless of whether the request is fulfilled. It reflects the impact of cache node distribution on the request path. The calculation method for APL has already been introduced in Section IV-A and will not be reiterated here.

Fig.6 provides a comprehensive illustration of the relationship between the average path length and the number of clients as the number of cache nodes varies. The graph shows a comparison of the results obtained from four different scenarios: a scenario with no cache nodes, and a scenario with 1, 5, and 10 cache nodes deployed in the positive region of the axis, respectively. It should be noted that, according to the principle of symmetrical configuration of 2 × n, a cache node corresponding to the positive region is also implicitly configured in the negative region of the axis. From this figure, it can be observed that when the number of clients is relatively low, the difference in the average path lengths observed in the four scenarios is insignificant. This finding is consistent with the actual situation, namely that increasing the number of cache nodes is not a crucial requirement for the case of fewer clients. Nevertheless, as the number of clients increases, the inclusion of cache nodes can markedly diminish the average path length for clients. However, when the number of clients exceeds 100, for a fixed number of cache nodes, the average path length does not continue to decrease as the number of clients increases, but rather reaches a stable value.

Furthermore, the figure illustrates that an increase in the number of cache nodes results in a reduction in the average path length. This is due to the fact that the deployment of high-capacity caches has the effect of significantly improving the cache hit rate, which in turn effectively reduces the number of client requests that bypass the cache in order to access the source node directly. As a result, network performance is optimized and improved. This optimization mechanism serves to both reduce the data transmission delay and diminish the load pressure on the source node, thereby enhancing the overall performance of the network. Fig.7 illustrates the trend of the average path length with respect to the duration of the simulation. In the figure, the dashed line represents the theoretical prediction, while the solid line depicts the actual simulation results. A detailed examination of the figure reveals that during the initial phase of the simulation, characterized by a relatively short simulation time, the high frequency of client requests within a limited time span facilitates data sharing among nodes. This enables nodes to efficiently cache and promptly respond to the same data requests from multiple clients, resulting in the actual observed average path lengths being considerably lower than the theoretically anticipated values. This phenomenon indicates that even in scenarios where nodes are not explicitly configured as caching nodes,

the NDN protocol can exhibit cache-like effects by merging requests for the same content.

As the duration of the simulation is increased, the distribution of client requests becomes increasingly decentralized, losing the highly concentrated characteristics that were initially observed. This results in a reduction in the overlap between requests and a weakening of the collaborative caching effect between nodes. At this juncture, it becomes imperative to rely on specialized caching mechanisms in order to maintain data availability and reduce access latency. Consequently, as the simulation time increases, the average path length approaches the theoretical value, demonstrating the pivotal function of caching nodes in enhancing network performance and reducing path length over extended time periods. In conclusion, when requests are concentrated, the role of specialized caching mechanisms is not significant due to the existence of natural caching effects. Conversely, when the distribution of requests becomes sparse, the application of caching mechanisms becomes particularly critical, as they can effectively reduce the average path length of the client, thus improving the overall efficiency and service quality of the network.

C. Average Route Hops

The Average Route Hops (ARH) refers to the average number of routing hops required to respond to user requests over a certain period of time. It reflects both the availability and redundancy of the cache, as well as the rationality of the cache placement. The more optimal the cache placement, the smaller the ARH, resulting in lower average user access latency. The ARH calculation formula is as follows:

$$ARH = \frac{1}{\sum_{i=1}^{M} \mathcal{I}_{u_i}} \sum_{i=1}^{M} \mathcal{I}_{u_i} \cdot h\left(u_i, c_k\right) \tag{21}$$

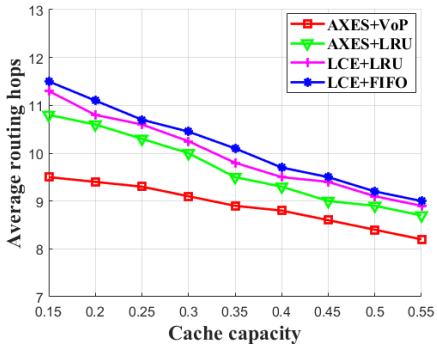


Fig. 8: Plot of average routing hops vs. cache capacity.

where $\mathcal{I}_{u_i}$ is the total number of interest requests made by user u_i, and $h\left(u_i, c_k\right)$ represents the number of routing hops required for the interest request from user u_i to be satisfied by cache node c_k. The effect of node cache capacity on the average number of routing hops required by the user to access the target content is shown in Fig.8. It can be seen that the average number of routing hops decreases significantly as the node cache capacity increases. This is because as the cache capacity increases, the nodes in the network are able to store more valid data, which in turn improves the local accessibility of the content. Specifically, with richer cache content, interest packets are more likely to get direct hits on the required data in the caches of nodes along the way as they are forwarded to the producer nodes, thus significantly reducing the number of hops for interest packets to traverse the network. Further analysis shows that among the caching strategies compared, the AXES+VoP strategy proposed in this paper has the lowest average number of routing hops, followed by the AXES+LRU, LCE+LRU and LCE+FIFO strategies. In the scenario where the cache capacity is set to 0.55, the average routing hop count of the AXES+VoP policy is only 8.1, which is reduced by 0.7, 0.9 and 0.94 hops respectively compared to the other three policies. This advantage is due to the fact that the AXES+VoP policy can dynamically optimise the cached content and effectively reduce the average latency of user requests by predicting and caching the most popular

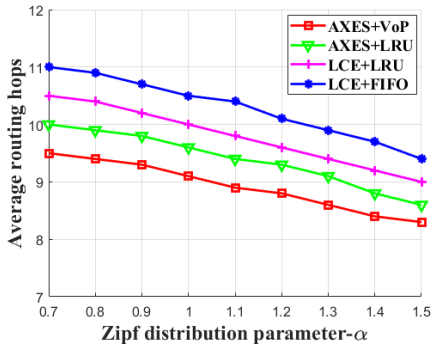


Fig. 9: Plot of average routing hops vs. Zipf distribution parameters.

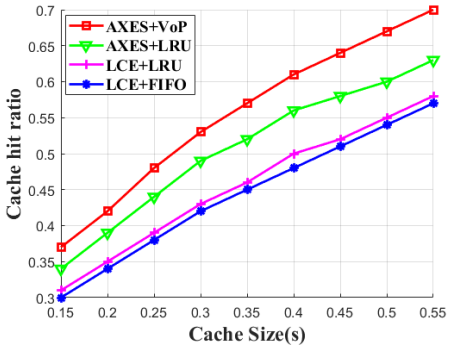


Fig. 10: The effect of cache capacity on the cache hit ratio.

content in advance, demonstrating the significant effectiveness of this policy in reducing the average routing hops and its practical value. In conclusion, the cache optimization strategy proposed in this paper is not only theoretically reasonable, but can also significantly improve the content distribution efficiency and reduce the network load in practical applications. Fig.9 provides insight into the specific impact of the Zipf distribution parameter on the average number of routing hops, confirming the effectiveness of the Zipf parameter as a tool for understanding and quantifying the phenomenon of centralisation of user data requests. The figure intuitively observes that as α increases, i.e. as the centralisation of user requests increases, the performance of each caching strategy generally shows an improving trend. This phenomenon reflects that the centralization of data requests helps the caching system to use cache resources more efficiently, thus improving overall performance. It is particularly noteworthy that the AXES+VoP cache optimization strategy proposed in this paper shows the best performance among all the caching strategies compared. In particular, in the case of a small Zipf parameter (e.g. $\alpha = 0.7$), the AXES+VoP policy is able to achieve a significant reduction in the average routing hops due to its unique optimisation mechanism, despite the fact that the randomness of the content requested by the user at this point in time is relatively high, making it difficult for the traditional caching policies to accurately predict the trend of future content requests. Specifically, under the condition that $\alpha = 0.7$, the average routing hop count of the AXES+VoP policy is only 9.5, which is a particularly obvious performance advantage compared to the other three policies (0.5, 1 and 1.5 hops lower, respectively). This result further confirms the effectiveness of the AXES+VoP policy in dealing with scenarios with high randomness of content requests.

D. Cache Hit Ratio

The cache hit ratio is measured as the proportion of requests satisfied by router nodes rather than content providers, thereby balancing content requests across the available cache resources. If a consumer can retrieve the requested content from a router node, the load on the content provider is significantly reduced. Thus, cache hit ratio is one of the key metrics for evaluating NDN performance. The formula for calculating the cache hit ratio is as follows:

$$CHR = \frac{\sum_{i=1}^{M} \mathcal{I}_{u_i}^{hit}}{\sum_{i=1}^{M} \mathcal{I}_{u_i}} \tag{22}$$

where $\mathcal{I}_{u_i}^{hit}$ represents the number of cache hits for user u_i. Fig.10 shows in detail the effect of the cache capacity size on the cache hit rate. From the figure, it can be clearly seen that the AXES+VoP policy proposed in this paper has the highest cache hit rate among all the caching policies compared, closely

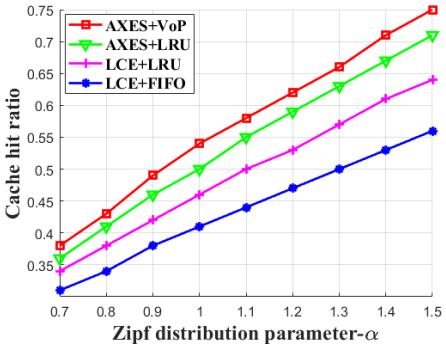


Fig. 11: The effect of the Zipf distribution parameters on the cache hit ratio.

followed by the AXES+LRU, LCE+LRU and LCE+FIFO policies. This result indicates that the AXES+VoP policy has significant advantages in improving cache efficiency.

Further analysis shows that as the cache capacity is gradually increased, the hit rates of all four caching strategies show an increasing trend. It is worth noting that the AXES+VoP strategy proposed in this paper still maintains good performance even in the context of relatively small cache capacities. This phenomenon is attributed to the fact that when cache resources are limited, the strategy is able to efficiently use the limited cache space and effectively reduce cache redundancy in the network by prioritising the storage of the more popular content, thus improving the responsiveness of the cache space to user requests. When the cache capacity is increased to a certain value (e.g. 0.55), the cache hit rate of the AXES+VoP policy reaches 0.7, which is a more significant performance benefit compared to the other three policies (0.04, 0.11 and 0.12 improvement respectively). This result not only confirms the stability and efficiency of the AXES+VoP policy when the cache capacity varies, but also highlights the potential of this policy to improve the overall performance of the caching system. Fig.11 gives an insight into the effect of the Zipf distribution parameters on the cache hit rate. The figure shows that the AXES+VoP policy proposed in this paper has the highest cache hit rate among all the policies. When the Zipf distribution parameter is small, the distribution of user-requested content is more dispersed, and at this point, it is difficult for the traditional caching scheme to effectively predict the content that may be requested in the future, resulting in a lower cache hit rate. As the Zipf distribution parameter increases, the content requested by users gradually becomes more concentrated. Following this trend, the hit rates of all four caching strategies improve. However, since the AXES+VoP strategy proposed in this paper is able to dynamically identify and cache highly popular content, this strategy shows the best performance in improving the cache hit rate. In particular, when the Zipf distribution parameter reaches 1.5, the AXES+VoP policy achieves a cache hit rate of 0.75, which is a particularly significant performance advantage over the other three policies (2%, 11% and 19% improvement respectively). This result further highlights the great potential of the policy proposed in this paper to improve the overall efficiency of the caching system and the user experience.

VI. Conclusion and Future Work

This paper explores the issue of cache optimization in LEO satellite networks and proposes a hybrid caching strategy based on NDN. By introducing a dynamic content VoP matrix and an axis-based cache placement method, we effectively enhance network cache efficiency and data transmission performance. Simulation results demonstrate that this strategy excels in reducing content transmission delay, optimizing path length, and improving cache hit rate. Despite the aforementioned improvements, the proposed approach still requires further investigation with regard to computational complexity and practical deployment feasibility. Future research will explore multi-layer cooperative caching strategies in satellite networks, leveraging collaboration among satellites at different orbital layers to enhance content accessibility. Moreover, predictive caching algorithms based on machine learning will be intro-

duced to further optimize real-time content placement and update decisions. Another key challenge lies in the high mobility of satellite nodes and the resulting dynamic changes in network topology. Therefore, future work needs to focus on developing more adaptive caching and forwarding mechanisms to improve the scalability and robustness of caching strategies in LEO satellite networks. By incorporating orbital prediction models and topology-aware caching and routing algorithms, the responsiveness and resilience of the caching system can be significantly improved. In addition, integrating the concept of Delay Tolerant Networking (DTN) can help alleviate transmission issues caused by intermittent connectivity.

REFERENCES

[1] O. Kodheli *et al.*, "Satellite Communications in the New Space Era: A Survey and Future Challenges," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 70–109, 2021.

[2] D. Zhang, S. Wang, J. Zhang, H. Zhu, T. Zhang, and X. Zheng, "A content distribution method of internet of vehicles based on edge cache and immune cloning strategy," *Ad Hoc Networks*, vol. 138, p. 103012, 2023.

[3] T. Liang, K. An, and S. Shi, "Statistical Modeling-Based Deployment Issue in Cognitive Satellite Terrestrial Networks," *IEEE Communications Letters*, vol. 21, no. 12, pp. 2726–2729, Dec. 2017.

[4] M. Liu and Y. Liu, "Price-Based Distributed Offloading for Mobile-Edge Computing with Computation Capacity Constraints," *IEEE Communications Letters*, vol. 22, no. 12, pp. 2647–2651, Dec. 2018.

[5] J. Wu, Z. Yang, C. Wu, Z. Li, and F. C. M. Lau, "Understanding the Usage Characteristics of Personalized Online Video Services," *IEEE Transactions on Multimedia*, vol. 21, no. 2, pp. 513–526, Feb. 2019.

[6] Y. Wang, C. Piao, C. H. Liu, C. Zhou, and J. Tang, "Modeling user interests with online social network influence by memory augmented sequence learning," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 541–554, Jan.–Mar. 2021.

[7] M. A. Naeem, T. N. Nguyen, R. Ali, K. Cengiz, Y. Meng, and T. Khurshaid, "Hybrid cache management in IoT-based named data networking," *IEEE Internet of Things Journal*, vol. 9, no. 10, pp. 7140–7150, May 2022.

[8] S. D'Oro, L. Galluccio, G. Morabito, and S. Palazzo, "SatCache: A profile-aware caching strategy for information-centric satellite networks," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 4, pp. 436–444, Apr. 2014.

[9] G. Zhong, J. Yan, L. Kuang, and "QoE-driven social-aware caching placement for terrestrial-satellite networks," *China Communications*, vol. 15, no. 10, pp. 60–72, Oct. 2018.

[10] L. Yang, X. Kong, Y. Qi, and C. Pan, "A collaborative cache strategy in satellite-ground integrated network based on multiaccess edge computing," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 8121509, 2021.

[11] X. Zhu, C. Jiang, L. Kuang, and Z. Zhao, "Cooperative multilayer edge caching in integrated satellite-terrestrial networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 5, pp. 2924–2937, May 2022.

[12] Z. Liu, W. Li, J. Feng, C. Pan, and Y. Li, "A regional interest-aware caching placement scheme for reducing latency in the LEO satellite networks," *Peer-to-Peer Networking and Applications*, vol. 15, Aug. 2022.

[13] M. Firouzjaee, K. Jamshidi, and N. Moghim, "A novel user preference-aware content caching algorithm in mobile edge networks," *Journal of Supercomputing*, vol. 80, pp. 1–24, 2024.

[14] M. Amadeo, G. Ruggeri, C. Campolo, and A. Molinaro, "Diversity-improved caching of popular transient contents in vehicular named data networking," *Computer Networks*, vol. 184, p. 107625, 2021.

[15] C. Bernardini, T. Silverston, and O. Festor, "MPC: Popularity-based caching strategy for content centric networks," in *Proc. IEEE International Conference on Communications (ICC)*, 2013, pp. 3619–3623.

[16] M. D. Ong, M. Chen, T. Taleb, X. Wang, and V. C. Leung, "FGPC: Fine-grained popularity-based caching design for content centric networking," in *Proc. 17th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2014, pp. 295–302.

[17] S. Zhang, J. Li, H. Luo, J. Gao, L. Zhao, and X. S. Shen, "Low-latency and fresh content provision in information-centric vehicular networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 5, pp. 1723–1738, May 2022.

[18] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "WAVE: Popularity-based and collaborative in-network caching for content-oriented networks," in *Proc. IEEE INFOCOM Workshops*, 2012, pp. 316–321.

[19] T. Zhang, Z. Yan, J. Duan, and Z. Zhang, "Information Forwarding Strategy of Internet of Vehicles in Named Data Network," *Computer and Modernization*, no. 11, pp. 19–27, 2024, doi: 10.3969/j.issn.1006-2475.2024.11.004.

[20] J. Zhang, Y. Yang, H. Sang, Z. Gao, and T. Song, "Content-Aware Proportional Caching for Efficient Data Delivery over Satellite Network," in *Proc. IEEE GLOBECOM*, Kuala Lumpur, Malaysia, 2023, pp. 4890–4895, doi: 10.1109/GLOBECOM54140.2023.10437961.

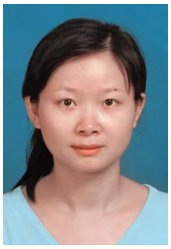
[21] S. Zhang, J. Li, H. Luo, J. Gao, L. Zhao, and X. S. Shen, "Low-latency and fresh content provision in information-centric vehicular networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 5, pp. 1723–1738, May 2022.

[22] Y. Liu, M. Chen, X. Wang, and V. C. M. Leung, "Edge content caching with popularity prediction using temporal features," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 14487–14498, Aug. 2022.

[23] M. Rodríguez-Pérez, S. Herrería-Alonso, A. Suárez-Gonzalez, J. C. López-Ardao, and R. Rodríguez-Rubio, "Cache placement in an NDN-based LEO satellite network constellation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 4, pp. 3579–3587, Aug. 2023.

[24] M. Coudron and S. Secci, "An implementation of Multipath TCP in ns-3," *Computer Networks*, vol. 116, pp. 1–11, 2017.

Yu Zhang received the Ph.D. degree in information and communication engineering from Southeast University, China, in 2020. Since January 2021, he has been a Lecturer with the School of Electronic & Information Engineering, Nanjing University of Information Science & Technology. From May 2014 to February 2015, he was a Baseband Software Development Engineer with ZTE Corporation. From December 2017 to December 2018, he was a Visiting Student with the University of Victoria, Canada. His current research interests include massive MIMO and millimeter-wave communications.



Yaqin Xie received the Ph.D. degree in communication and information systems from Southeast University, in 2011. She then joined the School of Electronic and Information Engineering, Nanjing University of Information Science and Technology. From February 2015 to February 2016, she was supported by the National Study Fund to visit King's College London, U.K., as a Visiting Scholar. Her current research interests include indoor localization, satellite navigation, routing planning, and artificial intelligence.

Jianyue Zhu received the B.S. degree in electrical and information engineering from Nanjing Agricultural University, Nanjing, China, in 2015. She received the Ph.D. degree from Southeast University, Nanjing, China, in 2021. Since April 2021, she has been a lecturer with the School of Information Science and Engineering, Nanjing University of Information Science and Technology, Nanjing, China. Her current research interests include non-orthogonal multiple access, intelligent reflecting surface, ultra-reliable and low latency communications, and optimization theory.

Chengshuai Zhou is currently pursuing a Master's degree in the School of Electronics and Information Engineering, Nanjing University of Information Science and Technology, Nanjing, China. His research interest is cache optimization and prefetching strategy.

Junmin Wu received the B.S. degree in radio technology from Southeast University, Nanjing, China, in 1994 and the M.S. degree in Electric Power System & Automation from Electric Power Automation Research Institute, Jiangsu, in 2000. As a Senior Engineer in State Grid Smart Grid Research Institute, China, his research interest includes the Deterministic Network, Industrial Ethernet, Wireless Communication.

Zhongyu Liu is currently pursuing the M.S. degree with Nanjing University of Information Science and Technology. His research interests include congestion control theory.