Journal of Scientific & Industrial Research
Vol. 82, February 2023, pp. 210-218
DOI: 10.56042/jsir.v82i2.70252

Network Path Optimization Strategy using Collaborative Cache for Delay Tolerant Networks

Chiranjeevi Kunamalla* & K Shahu Chatrapati
Department of computer science & engineering, JNTU Hyderabad 500 085, Telangana, India

Received 23 Aug 2022; revised 26 Sep 2022; accepted 07 Oct 2022

The data transmissions over Delay Tolerant Networks (DTN) and Social-based Opportunistic networks have increased in the last few years due to a higher demand for remote transmissions. The wide applications of DTN have significantly motivated the researchers to focus on finding optimized routing strategies by optimizing various parameters such as energy, cost and congestion. Nonetheless, these parallel research outcomes have reported few further bottlenecks to improve the strategies. Henceforth, this work proposes a novel approach to optimize the routing paths using the cache collaboration method. This proposed method identifies the data-sharing strategies and subsequently identifies the sub-set of the paths between the source and destinations. Further optimizes the path using standard measures such as cost, transmission speed, and the network traffic conditions; lastly Data-Centric and Cost Optimized Routing Path is Identification. This work results in a nearly 20% reduction in the distance between the nodes, a 15% reduction of time in path identification and a nearly 50% reduction in cache allocation demand over multiple iterations compared to the existing models.

Keywords: Cache collaboration, Cache discovery time, Data affinity, Mean load distance, Routing

Introduction

The popularity of micro videos has been on the rise thanks to the proliferation of social media websites and intelligent terminal devices. Traditional cloud computing cannot fulfil customers' low-latency video service needs owing to the massive number of smart devices attached to the core network, which strains the backhaul connection. The article recommends adopting proactive video caching at the edge to enhance end-user service quality. This study introduces Collaborative Video Caching (CVC) infrastructure. CVC outperforms more traditional caching algorithms for video services regarding cache hit rate and user waiting time. A collaborative Caching Decision Technique (CCD) and a Collaborative Service Response Method (CSRM) are supposed to enhance the hit rate and decrease delay with Collaborative Shared Resources (CSR). In order to anticipate future video demands and cut down on transmission costs, the study presents an algorithm based on federated learning.

The Content Delivery Network (CDN) concept, as proposed by Chen *et al.*[1], speed up and enhance network transmissions while decreasing their duration. The study uses machine learning and greedy algorithms to provide Edge Cooperative Caching (ECC). To begin, we use a neural collaborative filtering-based approach to foretell how well a piece of content would do. Afterwards, the greedy approach is used to maximize potential outcomes for content distribution inside the cooperative cache setting. The goal of using ECC is to shorten the time it takes to send data. The efficacy of the ECC is evaluated by computer simulation. According to tests, the ECC decreases average content transfer time, enhances cache hit rate, and better uses cache capacity.

Li *et al.*[2] demonstrated that CVC minimizes system communication and caching costs. An innovative method for 5G collaborative caching is proposed in this research. The proposed method uses the shared local resources of the 5G wireless hotspot network to increase data availability and access rate in the immediate area. Collaborative caching suffers from constraints such as sparse storage, remote locations, little visibility, and high volumes of material. Caching performance, local content availability, and transmission latency in a 5G wireless network hotspot are all enhanced when these constraints are considered. The proposed method uses machine learning algorithms, Zipf distribution, and knapsack to make the most of the hotspot cache's capacity and respond appropriately to user requests for which only

*Author for Correspondence
E-mail: jeeva.kunam@gmail.com

a subset of the necessary data is now available. Results from computer simulations indicate that the 5G wireless network design developed may improve transmission latency and hotspot hit ratio. In this paper we used an innovative strategy for further optimizing the routing pathways by using the cache cooperation technique and optimize various characteristics, including energy consumption, cost, and the amount of traffic congestion.

Research Reviews

Further, over the baseline methods, a good number of research attempts can be observed, and in this section of the work, the parallel research outcomes are critically analyzed.

Heterogeneous terahertz (THz) networks with caching capabilities alleviate the adverse effects of rising wireless video traffic and provide diverse video streaming options. In order to provide users with varying degrees of video quality, the authors propose a two-tier cooperative caching approach based on Scalable Video Coding (SVC). Important layer files are sent to users through cooperative cellular multicast-D2D and THz broadcasts. When combined with an efficient multicast transmission resource scheduling approach, a base-layer multicast group aggregation process may significantly boost resource utilization and spatial reuse. Different studies[3,4], have optimized an efficiency function by deriving formulas for the average increase in offloading capacity and the corresponding energy use. A long tail heuristic and a greedy gradient descent method both work to resolve the problem. The simulation results verify the two approaches, indicating that the suggested caching-transmission technique may achieve near-optimal performance compared to existing systems.[3]

MEC may be used to boost 5G QoE effectively. If the method can uncover concealed information about users' social media relationships, it will improve video caching. Within the scope of this study, Chiang *et al.*[5] introduced a Collaborative Social QoE-driven Video Caching and Adaptation (CSQCA) architecture. This research aimed to design and implement a two-tier MEC collaborative video caching architecture that stores frequently accessed films across a large number of edge servers. Second, our research suggests a proactive caching strategy that considers social interactions and social network video sharing. The last section discusses a quality-of-experience-driven video adaptation approach for edge servers to transcode locally-stored movies on demand. Reality-based information is the backbone of every good simulation. Experimental results reveal that the CSQCA framework outperforms baseline cache methods regarding hit rates and user satisfaction.

Sun *et al.*[6] analyze distributed content caching in a collaborative edge caching system in which a centralized hub broadcasts content migration information to all edge nodes. Each edge node is outfitted with its mini-base station and cache for quick data retrieval from the network. To improve content caching and teamwork, we design an online decision-making problem to boost the hit-to-miss ratio and guarantee a high-quality experience for the end user. Likewise, it is taken for granted that online data about the content's popularity would need to be collected regularly. Through the use of Thompson sampling for in-the-moment caching, this study introduces a decentralized approach to learning about the popularity of online content. The proposed method performed very well during simulations regarding the cache hit ratio and user experience.

According to Mehrabi *et al.*[7], multi-access edge computing can potentially centralize network edge access for mobile clients. The study develops a low-complexity self-tuned bitrate selection method and provides a cache replacement mechanism called retention-based collaborative caching. Collaborative caching boosts average video bitrate while decreasing data traffic when a certain BFTR threshold is reached. The results inform the development of a 5G collaborative caching strategy for mobile edge systems.

To lessen the strain on the 5G backhaul and enhance the user experience, wireless edge caching stations are being installed (ECSs). In this research, we look at the characteristics of edge caching systems operating across wireless networks. There is a set amount of time that a cached file will be in use, and new data will be sent randomly in response to user requests. Initially, content files may be divided into several coded packets for distributed collaborative editing. Although previous work decided which content to cache, this research focuses on how to distribute coded packets of material-to-be-cached amongst ECSs to speed up content downloads. This research proposes employing Lyapunov algorithms for stochastic collaborative content placement. The suggested method uses coded caching to take advantage of variations in material popularity across different regions. It allows for an almost ideal long-term caching performance without the need to forecast the arrival of new content, according to a simulation of a real-world YouTube video request trace conducted by Chen *et al.*[8], caching provide a significant performance boost compared to other benchmark strategies.

In order to reduce service processing delays in 5G networks, multi-access Mobile Edge Computing (MEC) has been identified as a crucial technology. By storing copies of frequently-accessed data close to its final consumers, MEC might improve the quality of their service. As a result, the latency is reduced, and the quality is improved. The limited storage space of MEC is far behind that of the cloud. Managing caches effectively is essential. To enhance MEC content caching, the research suggests and assesses Predictive Collaborative Replacement (PCR). The proposed replacement method performs better than Least Frequently Used (LFU), Last Recently Used (LRU) and Least Frequently/Recently Used (LFRU).[9] This is supported by theoretical considerations and empirical results on a real-world dataset (MovieLens20M), compared to those obtained using LSTM-based caching.

Typical multi-hop wireless networks are inefficient and wasteful of bandwidth. Distributed content caching is studied by Zheng *et al.*[10] to enhance collaborative relaying. This research demonstrates that the optimization of cache locations is convex. The numerical results show that the caching method is superior to traditional relaying regarding outage performance.

Multiple scenarios with unknown user preferences are taken into account by Liu *et al.*[11] Cache providers (CPs) make storage adjustments for users based on the data at hand to maximize the cache hit rate, taking into account the dynamic nature of mobile edge caching scenarios. This work presents a collaborative online Bayesian clustering caching approach for the CP to independently learn from the users' interactive cache hit data. The uncertainty in the latent number of user groups is described using the Dirichlet multinomial mixture (DMM) model, which is a Bayesian generative framework. In order to learn about user preferences and mapping, this research provides a dynamic clustering technique based on a collapsed Gibbs sampling method. The generated mappings are then used to guide cache selections, which are made using an innovative bandit mechanism with clusters of arms to accelerate learning. This research contrasts non-cluster long-term edge caching strategies with dynamic Bayesian clustering. The suggested method outperforms caching solutions without clustering, according to numerical results based on real-world data.

To reduce the time it takes to transport files when both backhaul bandwidth and cache space are limited, Wang *et al.*[12] describe a fractional dynamic caching technique. SBSs only save a subset of each file in their static memory; the rest is only accessed from dynamic memory when needed. By determining the optimal prefetching approach, or the size of the pre-fetched file section and dynamic storage, this research aims to decrease the mean time it takes for a file to be delivered to a user. The goal of this convex optimization problem is to minimize the amount of time it takes to send a file. This paper determines the best prefetching strategy to use while the wireless data rate remains constant. A heuristic prefetching technique is then developed utilizing the conditional average wireless rate. The research also estimates the heuristic prefetching approach without taking file popularity into account, which helps to smooth out popularity swings. In addition, the study delves into the logistics of putting the proposed caching approach for files of varied sizes into practice. Analysis of numerical data shows that the proposed technique improves backhaul efficiency and speeds up file delivery. Heuristic prefetching is a nearly optimal method.

In order to provide fast and reliable IT and cloud computing services, MEC makes use of RAN. Task offloading and data caching at Access Points (APs) may reduce backhaul load and content retransmissions. It is challenging to determine which MEC server should keep data and what data should be saved due to the variety of edge networks and the uneven distribution of their users. It is very important for ENs to optimize storage utilization while minimizing service delay and energy consumption. Task offloading deadline, AP cache capacity, and MEC server computing capabilities are all considered in Feng *et al.*[13] analysis of a two-tier MEC system that allows data caching and computation offloading to reduce UE network cost. It is an instance of MINLOP optimization. When we restrict our focus to only one optimization variable, we transform the problem into a task-offloading convex optimization problem. The method also uses dynamic programming (DP) to deal with cache placement. Then The CDCCO process is recommended to be iterated. According to simulations, CDCCO can improve network performance while decreasing expenses. Throughput is increased, latency is decreased, and traffic is centralized using edge caching.

Instead of the standard apriority theory, Zhao *et al.*[14] propose a posteriori caching technique that factors into user preferences and content popularity. This research aimed to determine the optimal caching strategy for a network with a mix of macro base stations, small base stations, and user terminals. By restating content placement as a 0-1 knapsack issue and using the lagrangian multipliers technique, this research can reduce transmission time while simultaneously increasing the local hit rate. Cache performance is optimized by considering users' request history, commonalities, and social connections. We optimize the caching approach into a low-complexity heuristic method by considering request probabilities and ideal copy amounts. The suggested cooperative caching strategy enhances hit rate and transmission delay over the baselines of previous algorithms in simulation studies.

Learning-based caching in SCNs under user uncertainty is investigated by Xu *et al.*[15] Reducing transmission latency over the long term may be achieved by optimizing cache placement in each SBS. The authors use MAMAB to represent sequential multi-agent decision-making. Instead of guessing user preference and optimizing it, the study presents MAMAB-based algorithms for directly learning cache strategy online in a stationary or non-stationary setting. The study recommends two collaborative MAMAB algorithms based on agents to guarantee static performance. A decentralized MAMAB without SBS coordination is proposed. The effectiveness of the proposed approaches was shown via simulation. Several cache configurations are described.

Edge caching in wireless networks saves data transfer time and space for frequently accessed content. Maddah-Ali and Niesen's suggested method of coded caching improves data transmission by broadcasting signals to several receivers simultaneously. The vast majority of automated caching methods assume complete library availability. Frequently, all users need from an app is some kind of location-based context. Ultra-high-definition maps may be made available to partially autonomous vehicles. This study presents a formalization of the coded caching problem using edge cache nodes for context-aware content. Wan *et al.*[16] look at a content server that stores files depending on their physical location.

As the Internet of Things (IoT) has become the backbone for domain applications, so needs IoT search engines that can scour several data sources in real-time. IoT search engines must process millions of queries, including location, time, and keywords per second. The caching paradigm in the cloud may be used in this research for routinely n-hop neighbor activity regions, which is why caching solutions are proposed in a collaborative edge-cloud computing architecture. This method uses a spatial-temporal-keyword index to estimate the relevance of keywords and the variability of travel times to compile the most sought data from n-hop neighboring regions. Scientists use the proposed STK tree to solve unusual situations efficiently. Extensive experiments with both real-world and synthetic data sets show that the solution proposed by Tang *et al.*[17]outperforms state-of-the-art approaches in terms of query time and message volume.

Using mobile computing, content caching at the edge of the network is an emerging technology (MEC). Zhou *et al.*[18] found that mobile devices record vast quantities of varied content from many users in various settings.

For 6G networks to provide quality experience assurances, sophisticated network traffic management is required. Inherently diverse computer architecture can only provide this. Researchers propose a distributed heterogeneous computing platform (HCP) for UAVs and terrestrial Base Stations (BSs) using caching and cooperative communication. Heterogeneous Base Stations (hgNBs) are provided by UAVs and RRHs on the ground in Fadlullah *et al.*[19] Considering traffic distribution, UE mobility, and localized content attractiveness, researchers propose a two-stage federated learning approach among UEs, UAVs/BSs, and HCP to anticipate content caching sites. Updates during off-hours (or "worknights") can cut down on unnecessary drills for federated learning. If the hypothetical 6G small cell has an HCP, it will learn the global model and then share it with UEs so that they may make use of edge intelligence. Numerical analysis is used to determine the feasibility of the proposal.

It would be inefficient to store the state data of real-world entities fully in the cloud because of the infinite number of entities and time-varying states in cyber-physical systems (CPSs). Zhang *et al.*[20] describe an edge cloud caching method for entity states in a combined entity to improve networking search applications. In order to mine the evolving rules of CPS entities from raw observation sequences, this research presents an entity state feature extraction technique. We devised a strategy for storing data on collaborating entities' states to boost CPS's search precision while cutting down on search latency and power consumption. First, entities with time-varying states are clustered together, and then the cache is built based on the state information of the entities in each cluster. The strategy's real-time and accurate capabilities have been verified via simulation.

In the context of edge computing (EC), edge servers at base stations provide nearby app users with access to processing power and data storage. In order to guarantee quick data retrieval, app providers may cache data on edge servers. To reduce data caching, data migration, and QoS penalties, Xia *et al.*[21] investigate collaborative caching in EC. The CEDC issue, or collaborative edge data caching, is NP-complete. The authors propose CEDC-O, a universal method for resolving CEDC. CEDC-O is a Lyapunov optimization-based approach that may be used online without the knowledge of the future and achieves near-optimal results. On a real-world dataset, CEDC-O outperforms four state-of-the-art methods.

Theoretical Considerations and Methodology

Foundational Method for Routing Path Optimization

After setting the initial context in the previous section of this work, the foundational baseline method for routing optimization is discussed. Assuming that the complete network configuration is N, consisting of the list of nodes, n[], and the paths, p[], between the nodes, which can be represented as,

$$N = < n[], p[] > \quad \dots (1)$$

Also, assuming that a total number of n nodes and m paths between the nodes construct the network. Thus, this relation can be formulated as,

$$n[] = < n_1, n_2, n_3, \dots n_n > \quad \dots (2)$$

And,

$$p[] = < p_1, p_2, p_3, \dots p_m > \quad \dots (3)$$

It is natural to realize that there are multiple paths between given two nodes as p_i and p_{i+1} between n_i and n_{i+1}. Thus, this can be formulated as,

$$p_i \Rightarrow n_j \leftrightarrow n_{j+1} \quad \dots (4)$$

And,

$$p_{i+1} \Rightarrow n_j \leftrightarrow n_{j+1} \quad \dots (5)$$

Thus, the baseline method for path optimization recommends considering two cases. The cases are furnished here.

Case – I: If the length of any given path is lesser than the other path in data propagation, then the shorter path must be selected. As

$$\begin{aligned} &Iff\ Length(p_i) < Length(p_{i+1}) \\ &Then,\ p[] \Leftarrow p_i \end{aligned} \quad \dots (6)$$

Case – II: If the paths connect transitive nodes, then paths without transitive nodes must be selected. As

$$\begin{aligned} &Iff\ p_i \Rightarrow n_j \leftrightarrow n_{j+1} \,\&\&\, p_{i+1} \Rightarrow n_{j+1} \leftrightarrow n_{j+2} \,\&\&\, p_{i+2} \Rightarrow n_j \leftrightarrow n_{j+2} \\ &Then,\ p[] \Leftarrow \{p_i, p_{i+2}\} \end{aligned} \quad \dots (7)$$

Hence, these are the two recommendations from the baseline method of routing path optimizations.

During the process of routing path optimization, the network routers also consider the cache coherences, which are discussed in the next section of this work.

Foundational Method for Cache Collaboration

Further, the cache collaboration-driven path optimization foundational baseline methods are discussed in this work section.

Assuming that the complete data in the network under processing is D[], each data item can be identified as d_i. Hence, this can be formulated for k number of elements as,

$$D[] = < d_1, d_2, d_3, \dots d_k > \quad \dots (8)$$

Continuing from Eq. (2), each node will contain some parts of the data based on the cache capacity, which can be generalized as,

$$d_i \rightarrow n_j \quad \dots (9)$$

(or)

$$d_{i+1} \rightarrow n_{j+1} \quad \dots (10)$$

Nonetheless, it can also be possible that the combination of these two data, as mentioned above, is part of another node. It can be represented as,

$$d_i \cup d_{i+1} \rightarrow n_{j+2} \quad \dots (11)$$

Thus, the network routing paths connecting X must be considered for further optimization. Henceforth, based on baseline methods, further the most recent research advancements are discussed in the next section of this work.

Problem Formulation – Mathematical Model

After the detailed analysis of the baseline methods and the recent improvements over the existing systems, in this section of the work, the persistent research problems are discussed.

Firstly, the analysis of the time complexity for the path analysis is calculated. Assuming that each node requires t1 time to build the adjacency matrix in the network and based on the previous assumption for n number of nodes, total time complexity T can be formulated as,

$$T = t1 * n * (n-1) \quad \ldots (12)$$

Further,

$$T = t1 * n^2 \quad \ldots (13)$$

Naturally, the time t1 is higher due to the higher distance between the nodes, and since, the analysis time is higher for a significant path for considering various factors such as transmission speed and consistency of the bandwidth, the Eq. (13) can be re-written as,

$$T = n^3 \mid t1 \approx n \quad \ldots (14)$$

This high processing time complexity must be reduced. Secondly, the major challenge in the existing systems is the capacity analysis of the cache for sharing the data. Continuing from Eq. (11), assuming that the capacity of the node is X as

$$cap(n_{j+2}) \rightarrow X \quad \ldots (15)$$

$$cap(d_i \cup d_{i+1}) > X \quad \ldots (16)$$

Then, the existing algorithms are bound to fail and create a much more complex situation. Henceforth, the proposed solution using the mathematical model is elaborated in the next section of this work.

Proposed Solutions

In this section of the work, the proposed solution is furnished. Continuing from Eq. (11), the proposed method tries to build the solution using co-existence logic. Assuming that the data item to be cached for routing optimization is D_X, and this data item is again a collection of multiple small data parts, then this can be formulated as,

$$D_X = < d_1, d_2, d_3, \ldots d_r > \quad \ldots (17)$$

Further, this proposed model recommends building the collection of nodes, nn[], which contain the part of the data as,

$$nn[] = \prod_{any\{D_X[]\}} n[] \quad \ldots (18)$$

Here assuming that the source and the destination node for the routing task are n^S and n^D, respectively, and the path between nodes are pp[], then the selection of the paths can be formulated as,

$$pp[] = n^S \leftrightarrow n^D \quad \ldots (19)$$

Further, the adjacency matrix, ppa[] or the network connection paths between the nodes containing the data must be calculated as,

$$ppa[] \leftarrow \prod_{nn[]} p[] \quad \ldots (20)$$

And, for the final optimized routing path, ppo[],

$$ppo[] \leftarrow \prod_{ppa[]} pp[] \quad \ldots (21)$$

Thus, the final routing matrix or path is obtained. Henceforth, the proposed algorithms are furnished in the next section of this work.

Proposed Algorithms and Frameworks

After the detailed analysis of the proposed mathematical model, the proposed algorithms are furnished in this section of the work.

Firstly, the Data Demand Oriented Source and Destination Identification (DDO-SDI) Algorithm is discussed.

Algorithm - I: Data Demand Oriented Source and Destination Identification (**DDO-SDI**) Algorithm

```
Input: Set of Nodes as N[]
Set of Paths as P[]
Output: Source Node as NS
Destination Node as ND
Process:
Update the node list as N[]
For each node in N[] as N[i]
If N[i].DataPackate.Type == "ACK"
Then, Count[i] = +1
Sort {Count[]}
Return NS as Count[MAX] and ND as Count[0]
```

Routing is only one of many factors that DTN systems have to consider. One of the first things to consider is how easily accessible data about potential connections may be accessed. In interstellar communications, for instance, it is not uncommon for a planet or moon to interrupt contact and great distances to slow down transmissions. However, physics principles make it feasible to foresee the future regarding when contacts will be accessible and for how long. Scheduled or predictable encounters are those that may reasonably be expected. However, in disaster recovery networks, the future position of communication entities, such as emergency personnel, may not be known. Intermittent or opportunistic encounters are just what they sound like.

Secondly, the Data Affinity Driven Cache Member Identification (DAD-CMI) Algorithm is discussed.

```
Algorithm - II: Data Affinity Driven Cache Member Identification (DAD-CMI) Algorithm
Input: Set of Nodes as N[]
Set of Data as D[], Searching Data as SD
Output: List of Data Available Nodes as NN[]
Process:
Update the data sources as D[]
For each data item in D[] as D[i]
If SD == Any(D[i])
Then, If D[i].Available.Source == N[j]
Then, Build the Collection as NN[k] = N[j]
Return NN[]
```

It would help if you considered whether or not mobile nodes may be used and which ones are mobile. Three main scenarios may be used to categorize the degree of network mobility. First, it is conceivable that there are not any movable objects at all. In this situation, the presence or absence of a contact is determined by the efficacy of the associated communication channel. For instance, planets and other big space objects might temporarily disrupt communication between nodes in interplanetary networks. Second, it is feasible that the network has some mobile nodes but not others. These nodes are used for their portability and are nicknamed "Data Mules." The challenge of fairly dividing data among these nodes is crucial in routing since they are the principal means of transitive communication between two non-neighboring nodes.

Third, the Data-Driven Routing Path Identification (DDR-PI) Algorithm is discussed.

Thirdly, it is feasible that all or almost all of the network's nodes are mobile. A routing protocol in this situation may not need to exhaust all available alternatives during contact chances since there will be more of them. When all nodes (often people and vehicles) in a network are mobile, like in a disaster recovery network, we have a mobile ad hoc network. Another case in point is a network that includes vehicles such as automobiles, lorries, and buses.

```
Algorithm - III: Data-Driven Routing Path Identification (DDR-PI) Algorithm
Input: Source Node as NS
Destination Node as ND
List of Data Available Nodes as NN[]
Output: Routing Paths as RP[]
Process:
Accept the list of nodes as NN[]
For each node in the collection NN[] as NN[i]
If NN[i].ID = NS.ID || NN[i].ID = ND.ID
Then, RP[j] = NN[i].Link
Return RP[]
```

Finally, the Data Centric and Cost Optimized Routing Path Identification (DC-CO-OR) Algorithm is discussed. The key factors are data transmission cost, speed and bandwidth utilization on the network.

Network resource availability is the third factor to think about. Many nodes, like mobile phones, have restrictions on their capacity for storing data, sending data, and staying power on. Buses on the road, for example, may have more freedom. This data may be used by routing protocols to optimize the flow of traffic and storage space for messages. The proposed framework is furnished in Fig. 1.

```
Algorithm - IV: Data Centric and Cost Optimized Routing Path Identification (DC-CO-OR) Algorithm
Input: Routing Paths as RP[]
Output: Optimized Routing Path as ORP
Process:
Update the routing matrix RP[]
For each link in RP[] as RP[i]
Sort{RP[i].Cost, High(RP[i].Speed), RP[i].Traffic}
Return ORP as RP[0]
```

Further, in the next section of this work, the obtained results are discussed.

Results and Discussion

The results obtained from this proposed framework are highly satisfactory and are discussed in this work section. Firstly, the experimental setup is furnished in Table 1. The experiment is carried out with 100 nodes

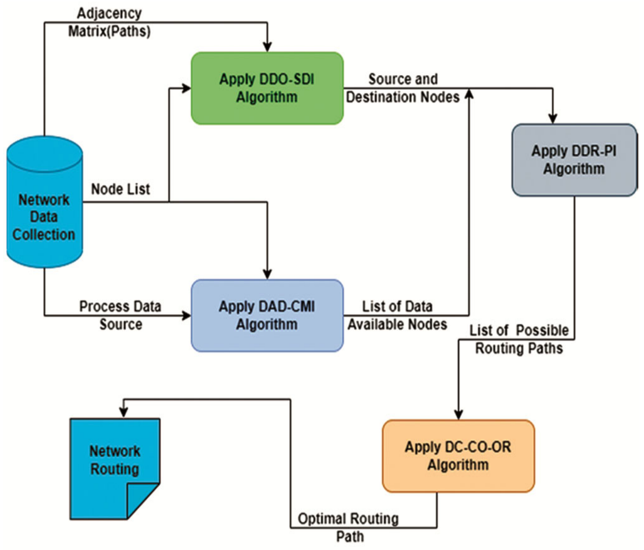

Fig. 1 — Proposed routing optimization framework

Table 1 — Experimental setup

Parameter	Value
Number of nodes in the network	100
Network area	1000 Km
Number of Iterations during Testing	100
Average Number of Nodes in the routing path	25
Average distance between the nodes	36 m

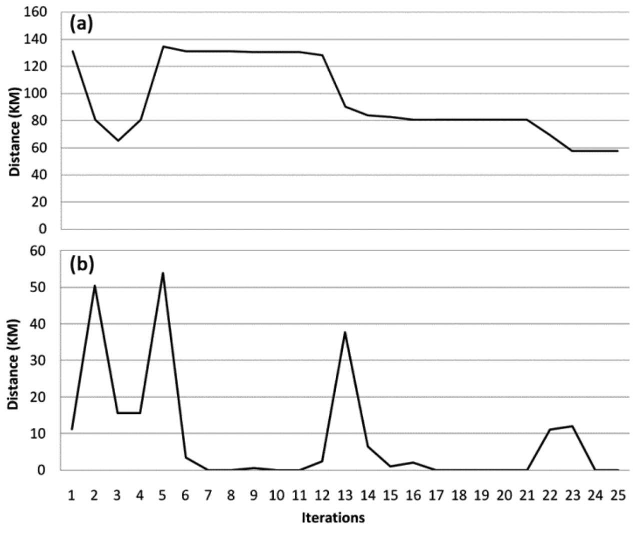

Fig. 2 — Distance Reduction Analysis: (a) Mean distance, (b) Iterative distance

and for 100 iterations using the proposed framework.

The mean distance between the nodes significantly reduces for every iteration, as shown in Fig. 2(a). The average distance between the nodes in 25 iterations is 95.52 Km.

Naturally, the distance between the nodes gradually decreases, which proves that path optimization is happening. The reduction in distance from iteration to iteration is also visualized in Fig. 2(b).

The average distance reduction between the nodes after 25 iterations is 8.94 KM. However, the reduction in distance is not only the performance improvement analysis. This work also proposed the improvement of the time complexity during the reduction process. The improvements in time complexity for the cache identification and formation are furnished in Table 2. It is natural to observe that the time to identify the cache collaboration is nearly zero after a few iterations.

The average time to identify the cache collaboration after 25 iterations is 0.00204 ns. Finally, this work also proposed reducing capacity issues

Table 2 — Cache collaboration time analysis

Iteration No (#)	Cache discovery time (ns)
1	0.016
2	0.016
3	0.015
4	0.004
5 to 25	0

Table 3 — Comparative analysis

Study	Model Complexity	Mean Distance Between Nodes (KM)	Mean Cache Allocation Size (KB)	Mean Time to Build Collaborative Cache (ns)
Chen *et al.*[1]	$O(n^2)$	120.35	20	0.00289
Sun & Nakhai [6]	$O(n^3)$	196.32	15	0.00587
Chen *et al.*[8]	$O(n^2)$	156.20	19	0.00368
Proposed Framework	$O(n)$	95.52	0.14	0.00204

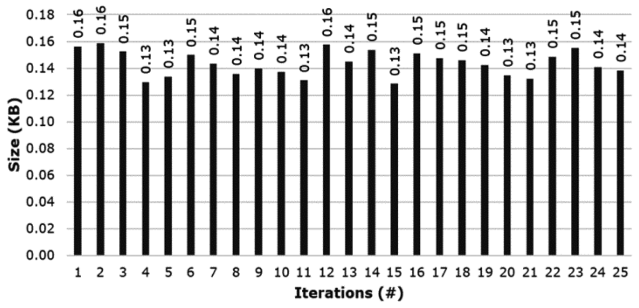

Fig. 3 — Cache allocation demand analysis

during the cache collaborations. The collective demand for building cache collaboration in terms of the space reduced has reduced over the iterations. The obtained results are also visualized graphically shown in Fig. 3. The average size of the cache configuration demand is 0.14 KB over 25 iterations. Henceforth, in this work's next section, the results are compared with the parallel benchmarked outcomes.

Comparative Analysis

The obtained results are discussed with the parallel research works in Table 3.

Hence, the proposed work has outperformed other parallel research outcomes.

Conclusions

This work is focused on discovering efficient routing algorithms by optimizing various characteristics, including energy consumption, cost, and the amount of traffic congestion. This study firstly applies the Data Demand Oriented Source and

Destination Identification (DDO-SDI) Algorithm to independently automate the source and destination identification process. Secondly, applies the Data Affinity Driven Cache Member Identification (DAD-CMI) Algorithm to identify the most suitable nodes, which already contain the data to be processed by the deployed applications, thirdly, applies the Data-Driven Routing Path Identification (DDR-PI) Algorithm to identify the possible paths between the source and destinations and finally, applies the Data Centric and Cost Optimized Routing Path Identification (DC-CO-OR) Algorithm to finalize the routing path. When compared to the models that are currently in use, the results of this work result in a reduction of approximately 20% of the distance between the nodes, a reduction of 15% of the time required for path identification and an approximate reduction of 50% of the demand for cache allocations across multiple iterations. The future work focuses on identifying the other metrics to improve Network Path Optimization.

References

1 Chen Y, Liu Y, Zhao J & Zhu Q, Mobile edge cache strategy based on neural collaborative filtering, *IEEE Access*, **8** (2020) 18475–18482.

2 Li Y, Hu S & Li G, CVC: A collaborative video caching framework based on federated learning at the edge, *IEEE Trans Netw Service Manag*, **19(2)** (2022) 1399–1412.

3 Furqan M, Zhang C, Yan W, Shahid A, Wasim M & Huang Y, A collaborative hotspot caching design for 5G cellular network, *IEEE Access*, **6** (2018) 38161–38170.

4 Li Q, Nayak A, Wang X, Wang D & Yu F R, A collaborative caching-transmission method for heterogeneous video services in cache-enabled terahertz heterogeneous networks, *IEEE Trans Veh Technol*, **71(3)** (2022) 3187–3200.

5 Chiang Y, Hsu C-H & Wei H-Y, Collaborative social-aware and qoe-driven video caching and adaptation in edge network, *IEEE Trans Multimed*, **23** (2021) 4311–4325.

6 Sun Z & Nakhai M R, Distributed learning-based cache replacement in collaborative edge networks, *IEEE Commun Lett*, **25(8)** (2021) 2669–2672.

7 Mehrabi A, Siekkinen M & Ylä-Jaaski A, QoE-traffic optimization through collaborative edge caching in adaptive mobile video streaming, *IEEE Access*, **6** (2018) 52261–52276.

8 Chen L, Song L, Chakareski J & Xu J, collaborative content placement among wireless edge caching stations with time-to-live cache, *IEEE Trans Multimed*, **22(2)** (2020) 432–444.

9 Ugwuanyi E E, Iqbal M & Dagiuklas T, A novel predictive-collaborative-replacement (PCR) intelligent caching scheme for multi-access edge computing, *IEEE Access*, **9** (2021) 37103–37115.

10 Zheng G, Suraweera H A & Krikidis I, Optimization of hybrid cache placement for collaborative relaying, *IEEE Commun Lett*, **21(2)** (2017) 442–445.

11 Liu J, Li D & Xu Y, Collaborative online edge caching with bayesian clustering in wireless networks, *IEEE IoT J*, **7(2)** (2020) 1548–1560.

12 Wang L & Zhou S, Fractional dynamic caching: A collaborative design of storage and backhaul, *IEEE Trans Veh Tech*, **69(4)** (2020) 4194–4206.

13 Feng H, Guo S, Yang L & Yang Y, Collaborative Data caching and computation offloading for multi-service mobile edge computing, *IEEE Trans Veh Technol*, **70(9)** (2021) 9408–9422.

14 Zhao X, Yuan P, li H & Tang S, Collaborative edge caching in context-aware device-to-device networks, *IEEE Trans Veh Technol*, **67(10)** (2018) 9583–9596.

15 Xu X, Tao M & Shen C, Collaborative multi-agent multi-armed bandit learning for small-cell caching, *IEEE Trans Wirel Commun*, **19(4)** (2020) 2570–2585.

16 Wan K, Cheng M, Kobayashi M & Caire G, On the optimal memory-load tradeoff of coded caching for location-based content, *IEEE Trans Commun*, **70(5)** (2022) 3047–3062.

17 Tang J, Zhou Z, Xue X & Wang G, Using collaborative edge-cloud cache for search in internet of things, *IEEE IoT J*, **7(2)** (2020) 922–936.

18 Zhou P, Gong S, Xu Z, Chen L, Xie Y, Jaing C & Ding X, Trustworthy and context-aware distributed online learning with autoscaling for content caching in collaborative mobile edge computing, *IEEE Trans Cogn Commun Netw*, **7(4)** (2021) 1032–1047.

19 Fadlullah Z & Kato N, HCP: Heterogeneous computing platform for federated learning based collaborative content caching towards 6g networks, *IEEE Trans Emerg Topics Comput*, **10(1)** (2022) 112–123.

20 Zhang P, Li X, Wu D & Wang R, Edge-cloud collaborative entity state data caching strategy toward networking search service in CPSs, *IEEE Trans Indust Info*, **17(10)** (2021) 6906–6915.

21 Xia X, Chen F, He Q, Grundy J, Abdelrazek M & Jin H, Online collaborative data caching in edge computing, *IEEE Trans Parallel Distrib Syst*, **32(2)** (2021) 281–294.