applied sciences

MDPI

Article

Efficient Caching for Data-Driven IoT Applications and Fast Content Delivery with Low Latency in ICN

Kamrul Hasan and Seong-Ho Jeong *

Department of Information and Communications Engineering, Hankuk University of Foreign Studies, 81, Oedae-ro, Mohyeon-eup, Cheoin-gu, Yongin-si, Gyeonggi-do 17035, Korea; kamrul@hufs.ac.kr

* Correspondence: shjeong@hufs.ac.kr; Tel.: +82-31-330-4642

Received: 31 August 2019; Accepted: 8 October 2019; Published: 6 November 2019

Abstract: Edge computing is a key paradigm for the various data-intensive Internet of Things (IoT) applications where caching plays a significant role at the edge of the network. This paradigm provides data-intensive services, computational activities, and application services to the proximity devices and end-users for fast content retrieval with a very low response time that fulfills the ultra-low latency goal of the 5G networks. Information-centric networking (ICN) is being acknowledged as an important technology for the fast content retrieval of multimedia content and content-based IoT applications. The main goal of ICN is to change the current location-dependent IP network architecture to location-independent and content-centric network architecture. ICN can fulfill the needs for caching to the vicinity of the edge devices without further storage deployment. In this paper, we propose an architecture for efficient caching at the edge devices for data-intensive IoT applications and a fast content access mechanism based on new clustering and caching procedures in ICN. The proposed cluster-based efficient caching mechanism provides the solution to the problem of the existing hash and on-path caching mechanisms, and the proposed content popularity mechanism increases the content availability at the proximity devices for reducing the content transfer time and packet loss ratio. We also provide the simulation results and mathematical analysis to prove that the proposed mechanism is better than other state-of-the-art caching mechanisms and the overall network efficiencies are increased.

Keywords: information-centric networking (ICN); edge computing (EC); Internet of Things (IoT); clustering and caching

1. Introduction

In recent years, a huge number of smart devices and sensors were deployed in the different areas of networks (e.g., smart health care system, vehicle to everything (V2X) communications, autonomous driving, industries, and smart home) to sense the real-time situation of the corresponding deployed environment and to collect the raw data [1,2]. Industrial Internet of Things (IIoT) (e.g., autonomous manufacturing systems, smart gas, grid, and oil systems) require a few millisecond latency between the smart sensors and the controller systems. Besides, augmented reality (AR), virtual reality (VR) applications, online gaming, body area network (BAN) [3] devices and a lot of other real-time application-oriented sensors may require a very low latency communication environment. Another alarming and important trend is the exponential increase of the IoT devices and the predicted data volume from those devices by 2020 will be 2.3 trillion gigabytes at each day by Mckinsey Global Institute. The data generated from the deployed smart devices and sensors are mostly unprocessed raw data to be processed before forwarding to the cloud network. Typically, the centralized cloud network is far away from end-user devices. Therefore, the consumed network bandwidth should

be increased for uploading a huge amount of unprocessed raw data and the use of memory in the cloud is also increased a lot accordingly. Moreover, the long distance between the end-users and the physical existence of the cloud networks may not be reliable for real-time services (e.g., live streaming, video conferencing). To overcome these issues, edge computing (EC) [4] has been introduced as an appropriate solution.

EC refers to the enabling technologies that allow computation to be performed at the edge of the network where "edge" indicates any path accompanying any computing and network resources between the cloud data centers and any data sources. For example, a smartphone is an edge device between smart home things and cloud; a small data center and a Cloudlet [5] is the edge between a cloud and a mobile device. EC pushes the end network devices to compute or preprocesses their produced data at the proximity of data sources. EC is also interchangeably used as Fog Computing (FC) [4], but EC emphasizes more towards the side of the thing, while FC emphasizes more on the infrastructure side. The importance of EC is to bring resources to the edge proximity devices such as storage, computation, and bandwidth. Therefore, these resources can facilitate the end-users to reduce the backbone data traffic and the response latency and to facilitate the data-driven IoT and IIoT applications.

Besides the latest advancement of the IIoT network and the increased data traffic from IoT and IIoT networks, the frequency of content delivery among the devices is increasing in a very fast manner, and inter-device communications are becoming an important issue. The Internet Protocol (IP) addresses have been dominating the current Internet architecture where the communication starts by a specific and location-oriented content request message. Due to the rapid growth of the Internet, it faces several problems, e.g., packet loss, slower download speed, network congestion and so on. To overcome these limitations, ICN has been proposed as a new network paradigm that is based on the content name instead of the current location-oriented communication mechanism. ICN is already becoming a very popular concept as a new Internet architecture for its significant improvements in network scalability, performance, and content delivery service with a reduce cost [6].

The combination of EC and ICN provides several directions [7,8] for solving the existing problems of the IoT network, data delivery and processing between the end-users and cloud including core and access networks. The current host-oriented network architecture is not suitable for the emerging low latency applications (e.g., augmented reality (AR) and virtual reality (VR) applications, autonomous driving, V2X security, and real-time gaming), although EC is deployed at the edge of the network because it works on downstream data for cloud services and upstream data for IoT services. Therefore, downstream services may not ensure low latency services. ICN has the caching capabilities at each device in the network where we can also use the computing resources based on the collaboration between the service providers and the end-users. In addition, ICN has other various characteristics that help the end-users to get their expected services. We will describe the benefits of combined EC and ICN mechanism in detail in Section 2.

The new characteristics of ICN accelerates it to become popular but there are some issues to be resolved, e.g., interest flooding, inefficient caching, naming, and so on. Basically, the objectives of ICN and EC are different, but there are some common functionalities (e.g., content caching). Therefore, the coexistence of them increases the network efficiencies in some scenarios (e.g., content delivery, content caching, preprocessing of the contents). Before discussing the integration of the EC and ICN concept, we first describe possible solutions to the existing ICN drawbacks and then we propose a combined platform based on EC and ICN.

The caching mechanisms used in the existing ICN and the EC mechanism could not handle the duplicate content caching in the nearby devices, and their content popularity mechanism works in individual devices. Our proposed unique caching mechanism collaborates with neighboring devices to handle the duplicate content in the nearby devices and maintain a global content popularity mechanism. Therefore, the popular content is cached to the ICN-enabled devices based on their global content popularity. They did not address the problem of content request packets flooding in their proposed

ICN and EC mechanism. We addressed this problem and provided the solution based on the clustering mechanism. In addition, the combination of ICN and EC increases the network computing and cache resources, and those help to increase the network efficiencies. In summary the following characteristics are the uniqueness of our proposed ICN and EC mechanism.

1. Reduction of the content access time based on the proposed efficient and unique caching mechanism.
2. The flooding of the content request messages is handled by the clustering mechanism and the network congestion is reduced in a certain dense network area.
3. Increase of the probability of content availability from the nearby content providers based on our unique caching mechanism.
4. Increase the bandwidth efficiency by preprocessing of the contents before uploading to the content servers or clouds and control of the flooding of content request messages.

The rest of the paper is organized as follows. Section 2 explains the motivation and background analysis of the integration of the EC and ICN concepts. Section 3 provides related studies on EC, ICN, combined EC and ICN strategies and mechanisms. Section 4 proposes an architecture for IoT and multimedia data handling based on clustering in ICN. Section 5 presents performance analysis. Finally, Section 6 concludes the paper with a summary.

2. Motivation and Background Analysis

2.1. Edge Computing

Edge computing is a model where the important data are processed by bringing the computational resources and data storages closer to the nearby required location of a network to reduce the response time, reduce the traffic volume in the core network and optimize the consumed local and global bandwidth. EC pushes the computational resources and data storages to the network edges instead of the centralized cloud management system. The important and larger volume of IoT data is preprocessed before sending to the global cloud data center and storing locally to access the important data within a very short time.

2.1.1. Classification of Edge Computing

The centralized Cloud computing (CC) [9] is unable to provide quality service to the end-users due to the enormous traffic load. Therefore, different solutions were proposed to decentralize the network traffic and to support the computational services to the edge of the network. There are several categories of EC based on the network structure, service types, and traffic of a specific area of the network. Figure 1 shows the classification of EC. Mobile Cloud computing (MCC) [10,11] is one of the EC paradigms, and it offers an offloading mechanism for the mobile devices into a combined system based on the integration of mobile Internet, CC, and mobile computing. Offloading a task to the cloud servers from a mobile user was the major goal to overcome the source mobile devices computational and storage resources limitations. Moreover, the lifetime of a battery of the mobile user is also increased due to the offloading of the task.

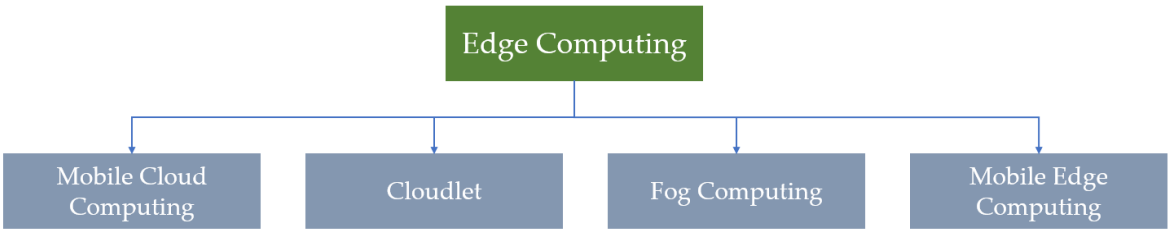


Figure 1. Types of edge computing.

As the extension of MCC, Cloudlet [12] was introduced as a type of EC, and it is reflected as the key enabling technologies for MCC. Although the concept of MCC was working fine, the distance between the source of mobile devices for offloading a task and the cloud server is very long. Therefore, the communication latency is very high, and the real-time task requires low latency that is not possible based on MCC. As a result, the task can be offloaded into the Cloudlet instead of the cloud servers to meet the lower latency communication and to provide the computation power to the end-users. The deployment of Cloudlet is considered to the nearby mobile devices with single hop proximity and it works as a virtual machine (VM).

FC [13,14] is another type of EC where the preprocessing happens before sending to the cloud servers. FC also provides other computational and resource-oriented services (e.g., offloading a task, caching, location awareness, and mobility information). European Telecommunications Standards Institute (ETSI) proposed the concept of mobile edge computing (MEC) [15,16], aiming to reduce communication latency and providing the location awareness services to the mobile users. ETSI also ensured that the requirements of EC will be fulfilled by the 5G mobile networks and beyond.

2.1.2. Why Edge Computing is Required?

The future network architecture will be service-oriented, and the contents of the user should be kept closer to the user, so that they can access their required content within a very short time. A lot of IoT devices were already deployed and the data generated from the deployed IoT devices will be 2.3 trillion gigabytes at each day. The generated data is not all useful, or some data need to be preprocessed or cached before sending to the cloud server. Therefore, a local processing mechanism is required to handle a huge amount of generated IoT data per day. The EC can provide all the requirements as we discussed before. Besides, we describe the following key reasons for using the EC to satisfy the end-users and for the efficient traffic and bandwidth management.

(1). Low Latency

Autonomous driving, AR, VR, and other real-time applications need very low latency communication to fulfill their goals. The current network architecture and even the current centralized cloud architecture cannot fulfill the goals of the real-time delay-sensitive applications. In many cases, EC can fulfill the expectations of real-time and delay-sensitive applications and; therefore, the quality of services (QoS) for the end-users will be increased.

(2). Traffic Management

The current network traffic is increasing towards some specific paths for a certain specific service provider due to the centralized cloud network architecture. Higher traffic flows create network congestion and packet loss due to insufficient bandwidth for a network. By considering the trillions of gigabytes of IoT data per day, the centralized cloud network will be unable to handle the huge amounts of contents. Therefore, the traffic should be handled nearby from the billions of source devices to optimize the bandwidth utilization and to reduce the traffic in the core networks. As a result, the EC paradigm is an appropriate candidate to play a meaningful role in traffic reduction on the core network.

(3). Scalability

The deployed IoT and end-user devices are increasing every day, and it is predicted that the number of IoT and the end-user devices will be millions and billions, which may create a scalability challenge soon. These millions and billions of devices will generate a huge amount of local traffic that will create congestion in the data centers or cloud centers. Therefore, the local edge servers can handle the huge amount of data in a decentralized manner. If the data flow is increased enormously towards a certain local edge server and the local edge server is unable to handle the data flow, then it can distribute the data flow towards the proximity local edge servers. Therefore, scalability challenge can be handled by distributed local edge servers.

2.2. Information-Centric Networking

The current host-oriented Internet architecture has several drawbacks (e.g., the limited IP address to identify a lot of IoT and other devices, location-dependent content access, and centralized architecture). To overcome these drawbacks, information-centric networking (ICN) [6] was proposed as a new Internet architecture. ICN is a location-independent and content-oriented Internet architecture where an end-user accesses the contents based on the content name instead of content location address. ICN has several characteristics such as in-network content caching, naming, packet-level security, etc., which will be useful to overcome the drawbacks of the existing Internet. ICN is an important and effective candidate for modern communication architecture including IoT and massive machine-type communication in the 5G era.

2.2.1. Types of ICN

ICN is an umbrella concept, and there are several types of proposals (e.g., Content Centric Networking (CCN), Named Data Networking (NDN), Data Oriented Network Architecture (DONA), Publish-Subscribe Internet Technology (PURSUIT), Publish-Subscribe Internet Routing Paradigm (PSIRP), Scalable & Adaptive Internet soLutions (SAIL), Architecture and Design for the Future Internet (4WARD), COntent Mediator architecture for content aware nETworks (COMET), CONVERGENCE, Mobility First) [6] continuously working under this umbrella concept. Among them, CCN [17] and NDN [18] are very familiar to the research arena, and recently both concepts have been merged and are working together. Although the core concept of all these types of ICN is the same, there are some little differences in routing, naming mechanism, name resolution handling, etc. mentioned in [6].

2.2.2. Communication Procedure in ICN

In this paper, we focus on CCN, a representative technology to explain the ICN concept. CCN and NDN are getting much of the research interest and the concepts of CCN and NDN are almost similar except for the mechanism of forwarding the content request packet. There are two types of packets proposed by CCN and NDN for content request and reply named as interest packet and data packet. The CCN forwarding engine decides the next destination for sending the interest packet. Figure 2 shows the CCN communication mechanism in details between an end-user and a content provider.

The CCN interest packet contains the content name instead of the content location address. The core network of the CCN is responsible for locating the data packet. The data packet contains some security-related information including required data that ensure the packet-level security in the CCN. CCN uses some key management systems and hash algorithms to secure their packet as well as the user. The CCN forwarding engine has three types of data structures: (1) Content store (CS), which is used for caching the incoming or outgoing content that is not available in the CS; (2) pending interest table (PIT), which is used to track the incoming interest packet including the face ID that helps to identify the individual user's path; (3) forwarding information base (FIB), which is used for selecting the appropriate outgoing face for the incoming interest packet. Following the above data structure in the CCN forwarding engine at each CCN node, the interest and data packets are handled in the CCN network architecture to send and receive the expected interest and data packets.

2.3. Reason for Coexistance of ICN and Edge Computing

We already described the features of ICN and edge computing. The individual characteristics of both ICN and EC are not fully suitable for respective scenarios. For example, EC provides the caching mechanism in the edge of the end-users but ICN provides caching mechanism at each node in the network. Therefore, the availability of the cached content will be increased automatically. To handle the millions and billions of IoT devices, EC does not provide any unique solution but the ICN naming technique can solve the IoT naming issues. Similarly, the coexistence of ICN and EC will be a great

environment for the future service-oriented network architecture that may solve both upstream and downstream content management issues.

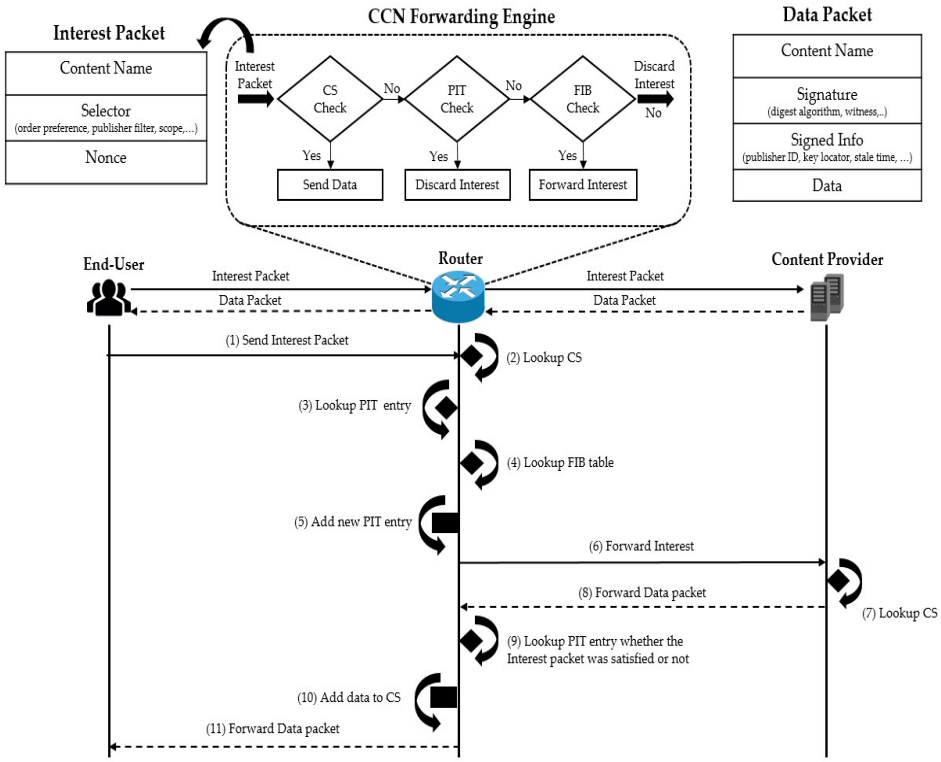


Figure 2. CCN communication mechanism.

2.3.1. Handling Duplicate Content Request

ICN can handle the duplicate content request by adding the incoming face in the PIT entry for the same content request packet where EC cannot handle this issue. Therefore, multiple content request messages will be forwarded to the same direction in the current EC environment. But, the coexistence of ICN and EC handle this issue effectively.

2.3.2. In-Network Caching Capability

EC has limited caching capability that can be used during the preprocessing of the IoT devices generated data for upstream data flow from the source IoT devices to the cloud servers. But, the real-time application-oriented downstream data flow for the general users (e.g., real-time video streaming services, AR and VR applications) must be cached in the proximity devices, which is not practical usually for a huge number of receivers. To support the huge number of end-user subscribers for the real-time applications, the coexistence of ICN and EC may provide them with the expected service.

2.3.3. Naming Procedure

The identification of the millions and billions of IoT devices is only possible by using the naming mechanism of ICN. In some cases, EC can handle some scenario for IoT where the number of IoT devices is not so high. For example, to recognize the IIoT areas devices, the naming mechanism will be a more appropriate solution.

2.3.4. Packet Level Security

Security is a very important aspect for the IoT, V2X, autonomous driving, and other important real-time applications and data sets. ICN provides the packet-level security, whereas EC needs extra

security mechanism to secure the communication between the devices and end-users. Therefore, the coexistence of ICN and EC do not need any extra security mechanism except for the very high-level security requirements.

2.3.5. Easier Mobility Management

The concept of ICN is content-oriented, while the current Internet architecture is host-oriented. The host-oriented architecture needs end-to-end connection establishment after handover, but ICN does not need that. Therefore, ICN provides better mobility management than the existing mechanisms because contents can be accessed directly without establishing the end-to-end connection again.

3. Related Work

Edge computing plays an important role at the edge of IoT networks in different ways, e.g., providing computing resources or caching resources to the end-users. Cloudlets are an effective solution for reducing the response time between IoT users, but the excessive workload increases the response time and computation delay, and sometimes the incoming request is unbearable. To solve the issue, an application-aware workload allocation model [19] was proposed to adjust the computing resources dynamically for different applications in each Cloudlet based on their workloads. An intensive analysis and a new proposal about the MEC framework from an air–ground integration perspective [20] have been done for IoT environment. They presented a real-world topology to show the performance improvements in computation capability and communication connectivity. Their work was proposed based on the software-defined networking (SDN) and centralized network architecture using Cloudlets in the edge of the network. The resources of the Cloudlets are not enough to provide the demands of the users of the dense network where massive connectivity is considered. The combination of the ICN-enabled network resources and end-users' resources is the probable solution that is proposed in this paper. FC was applied in the vehicular ad hoc network (VANET) [21] for efficient communication, location-aware service provision, improved response time, and lower latency. Here, cache size impacts on the cache hit ratio and the performance could be upgradeable by the integration of ICN and EC. Video processing [22] was also proposed in the edge of the multimedia IoT system that needs huge computing power and memory for real-time video processing. ICN can provide enough memory to the nearby location of the multimedia IoT system where the video is processed. Therefore, the coexistence of ICN and EC will be beneficial for the IoT devices and the real-time service consumers.

There are three different types of caching mechanisms [23] for ICN, and they are on-path caching, off-path caching [24], and hybrid caching technique. The most familiar and important characteristic of ICN is in-networking caching. In ICN, the end-users retrieve a content using a content request message (e.g., interest packet in CCN, a type of ICN) and receive the content using a replied content packet (e.g., data packet in CCN). The data packet is forwarded to the reverse path of the interest packet and all the traversed devices cache the incoming data packets. As a result, duplicate content caching occurs in the nearby devices that increase the memory used in the network, increases the content management time, and so on. A lot of researches on caching are already done, and some researches still are ongoing to increase the efficiency of the cache memory in the network.

The procedure of the on-path caching mechanism was studied in [25] and the related cache replacement algorithms were analyzed in [26,27]. In ICN, the data packets are cached in all the on-path traversed nodes or a set of on-path traversed nodes [28,29]. The other on-path caching mechanisms are studied, e.g., leave copy down (LCD) in [25], centrality-based caching in [28], ProbCache in [29], and leave copy everywhere (LCE) in [30]. Among the on-path caching mechanisms, popularity-driven [31] and content location-based [32] caching mechanisms become the most popular caching mechanism. The on-path caching mechanisms increase the content's cache hit ratio in some cases but at the same time, the redundant uses of the cache memory reduce the overall efficiency of the cache strategy of ICN.

In ICN, off-path caching is a technique of moving the cached content in the external deployed memory in the proximity of on-path network devices. The interest packet is forwarded to the content

provider and the replied data packet is forwarded in the reverse path of the interest packet. The user requests are handled by the core network entity and the impact of caching in the current network [33] and ICN environment [34]. There are several caching mechanisms were proposed based on the ICN concept. A caching mechanism was proposed named as "Breadcrumbs" [35], where they used some metadata related to cached content. These metadata help the end-users to find out the relevant and appropriate content from the network. The hash-routing mechanism was proposed as a mapping mechanism and later five different types of hash-routing techniques were applied in the ICN caching mechanism to show the impact of these strategies [24]. These caching mechanisms increase the content availability from the nearby content providers, but the efficiency of the memory usage is reduced a lot because of redundant content.

The combination of on-path and off-path caching is known as hybrid techniques. Scalable content routing for content-aware networking (SCAN) [36] is a hybrid caching technique where the bloom filter was used to cache the useful content and on-demand delivery of the cached contents. They compared their routing mechanism with the IP routing mechanism and showed that content transfer time was reduced compared to the existing Internet architecture. But, the combination of ICN and EC mechanisms provides faster data transfer environments than the hybrid SCAN mechanism.

The coexistence of ICN and EC is not a new area of research. Several works already being done to get the mix features outcomes into a single architecture. A fog-enabled edge learning mechanism was proposed [7] to design a cognitive CCN in the 5G environment to associatively learn and control the states of edge devices and the network resources. They used the caching capabilities and other network resources of edge devices. It ensures that the edge device network resources are useable for the betterment of network performance. Other works showed that collaborative edge caching [8] enable mobile users to fetch the contents from cache servers so that the user experience will be improved, and end-to-end communication latency will be reduced. These works indicate the coexistence of EC and ICN will improve the user experience and reduce the end-to-end communication latency based on the combined characteristics of the ICN and EC.

4. An Architecture for IoT and Multimedia Data Handling

The main goal of our proposal is to reduce the communication end-to-end latency, optimize data traffic in the core network, improve the content access time, provide a proper caching mechanism in the data-driven IoT applications, and support the fast delivery of multimedia content, AR, VR applications, and all the real-time streaming services based on the combined architecture of EC and ICN. To obtain our goal, we designed an architecture based on the integration of the features of EC and ICN. We consider multiple applicable scenarios (e.g., IoT, IIoT, V2X, autonomous deriving, AR, and VR). The service-oriented applications get the services from the service providers and their generated data are effectively handled by our proposed architecture. Figure 3 shows the overall view of our proposed architecture. The types of EC are deployed at the edge of the core and access network. The core network consists of several types of devices such as router and gateway, and the content providers are located at the edge of the core network. The global cloud servers are located in the centralized data center for different service providers and they are connected to the core network. Usually, the end-users access the content from the centralized data servers through the core network. But all the network devices and the end-users including content producers and consumers are enabled with ICN capabilities. Therefore, they can communicate with each other according to the ICN concept. Any devices in the network could be a content provider and a consumer in the ICN-enabled network architecture.

We divide our architecture into two scenarios: (1) Downstream data flow management based on the combined characteristics of EC and ICN, (2) upstream data handling based on the combined network resources of EC and ICN. The data flow means that a consumer or an end-user can access a content from any content sources (e.g., centralized cloud server, location-oriented content server, or any content service providers). The content requester only requests a content by using a content name instead of the location of the centralized cloud server, location-oriented content server, or any

content service providers. Therefore, the network devices are responsible for communicating between the content requester and the content provider. In our proposed architecture, any devices of the network could be a content provider because the ICN concept is enabled at each device in the network architecture. To get a clear understanding of our mechanism, a Tier architecture is represented in Figure 4. There are four Tiers in the architecture. The first Tier includes the end-users, all the sensor devices from IoT and IIoT applications, autonomous cars and industries, V2X communication, AR, VR applications, real-time monitoring devices, human wearable sensors as the content generators, and different types of consumers for various services. Tier 1 devices are also defined as the access network devices because they access the other Tiers' devices to communicate with other devices including Tier 1 devices. The same Tiers' devices can communicate directly (e.g., device to device (D2D), machine to machine (M2M), or using PC5 interface of cellular networks). These devices are classified into three categories:

1. Content generators only;
2. Content consumers only;
3. Content generator and consumer.

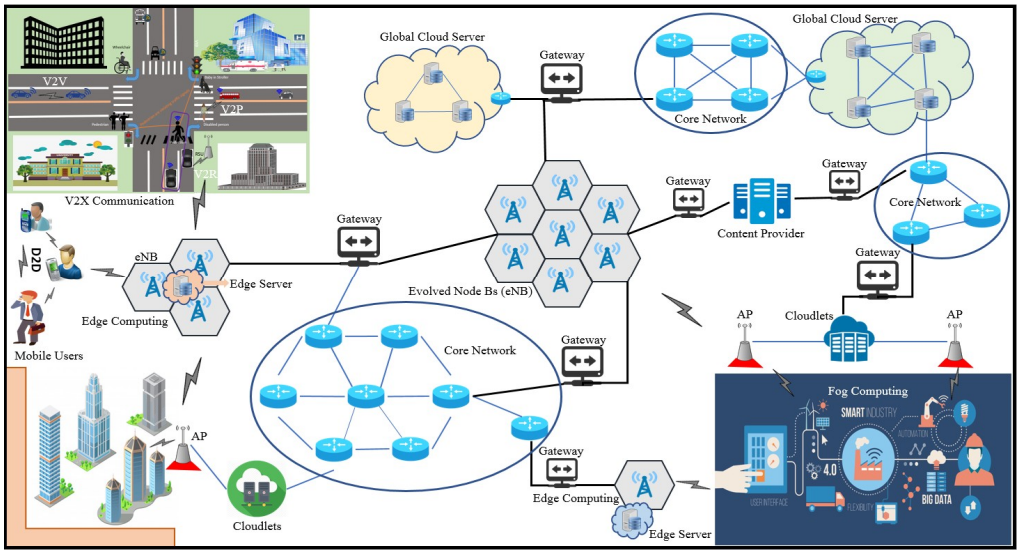


Figure 3. An overall view of our proposed architecture.

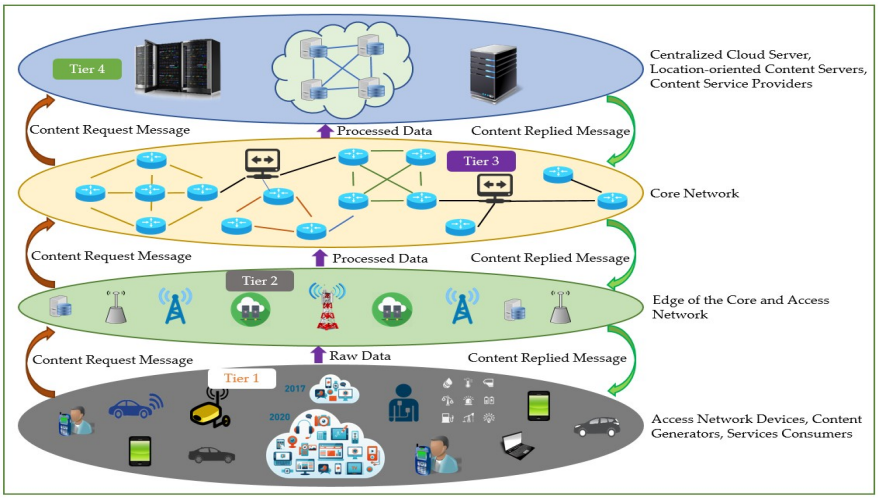


Figure 4. Content flow based on a tier architecture.

Content generators are only generating the content and those contents are used locally for different purposes (e.g., smart home services and industrial automation) and are stored in the global cloud server for future use (e.g., temperature sensor data and accident-related data). They never send any content request message to the network. They do not create any network traffic in the downstream data traffic. The generated data need to be collected and then some preprocessing is required before sending to the global cloud data center. Some devices of Tier 1 only consume the contents from different service providers. They never produce any data and the service providers always try to provide their best services to maximize the QoS of the services consumers. The maximum downstream data are used by the content consumers. The third category devices generate and consume data and services. They use both upstream and downstream data.

The huge number of end-users and IoT devices are continuously generating and using the generated data instantly, but the huge amount of data needs to be stored in the global cloud data center. The data need to be preprocessed before sending to the global cloud data center. Therefore, we apply EC mechanisms at the edge of the access and core network. Tier 2 shows the devices and the required types of EC mechanisms. The generated data from Tier 1 devices are preprocessed in Tier 2, and Tier 2 contains base stations, access points, and a cache server to support the EC mechanisms. The computational mechanism is applied to the sensor-generated data to reduce the volume of data. We do not mention the computational mechanism because the types of the computation mechanism depend on the objective of the service providers (e.g., the object identification mechanism may apply to the real-time video monitoring sensors or cameras). Similarly, appending the small-sized IoT generated data into a large file based on the sources and sensor types of a certain region. EC appends the small packets for the same destination. Therefore, the network traffic and the use of network resources are optimized.

A lot of network resources are available in the Tier 3 devices. The most important resources are the caching capabilities of every router, and their content request message-forwarding mechanism is another level of uniqueness in our proposed architecture. As a result, each router decides caching, forwarding, duplicate content request messages handling. Furthermore, to manage the downstream content flow and data traffic, these router's caching capabilities help the end-users to get their expected content from the nearby cached devices and reduce the communication latencies and enrich the QoS of the end-users. Although the existing caching mechanisms are inefficient for the optimal use of in-network caching capabilities, its use optimizes the network efficiencies and throughput. We describe our cluster-based in-network caching mechanism for the core network to increase the efficiencies of the core network. The clustering mechanism applied to the core network and the remaining Tier architecture will be the same.

Content is the salient target for the Tier 4 and the organization of these contents depends on the different service providers. The centralized content server is the pivotal architecture for most of the service providers, but nowadays distributed architecture is becoming an important architecture in order to satisfy the end-users by providing low-latency communications. Although the decentralized architecture increases the end user's satisfaction, they are location-oriented service providers. Therefore, network traffic flows in one way, creates network congestion, and increases the consumption of network bandwidth. Our in-network caching mechanism is distributed and all the devices from all Tiers cache the content based on the content popularity and importance of the content based on the content access time and location.

4.1. Downstream Data Flow Management

The downstream data is handled by the proper caching management in the traversed network devices. The end-users always need the requested content within a shorter time. Therefore, our goal is to keep the expected user data towards the edge of the users. We have more caching capabilities in our EC devices that are used for caching the most popular content for the nearby end-users. The efficient in-network caching mechanism is used for the more popular content caching and the less popular

contents are cached in the remote caching devices or the servers of the content service providers. The content popularity measurement procedure is described in the Section 4.1.1. The clustering mechanism is applied to the core network to handle the downstream data. The clustered network solves the content request message flooding issue and it provides an efficient routing mechanism. The core network is represented by a graph (G). Individually, the devices of the graph are denoted as a node. Algorithm 1 is used to make a cluster-based network and it provides us with a set of clusters. The cluster list, C_List, is used to make the cluster head for each cluster. In a cluster, each node is represented as a cluster node and the cluster head knows all the information (e.g., the available cache memory and the total cache memory of each node, available cached content at each node within the cluster, etc.) of the cluster node including the neighboring cluster head information. The content request messages handling mechanism is also changed at each cluster node. Section 4.1.4 covers the content access mechanism.

Algorithm 1: An algorithm to make clusters (G).

Input: All the nodes of the core network a graph, G

Output: A set of clusters

```
Definition of used variables:
Ck = kth Cluster;
Neighbor(n) = Include all the neighbors of node n;
FINISH(n) = It indicates whether the node n is included in a cluster or not.

Initialization:
For node 1 to N, FINISH(n) = FALSE; N represents the total number of nodes.
C_List = Ø;
k = 1;

Begin;
Randomly choose a node p among N nodes from Graph G
    For each node q in Neighbor(p)
        if q ∉ Ck
            Ck = Ck ∪ {q};
            FINISH(q) = TRUE;
        else
            if Neighbor(p) = Ø
                Ck = Ck ∪ {p};
                FINISH(p) = TRUE;
            else
                continue;
    C_List = C_List ∪ Ck;
    k = k + 1;
  while (n in 1 to N, FINISH(n) = TRUE)

End;
return C_List;
```

4.1.1. Content Popularity Measurements

In ICN, content is transferred as chunks. Every chunk requires a content request packet and content are replied as a sequence of chunks. This content chunk is cached into the CS and cache replacement policy is required when the memory is full. We already mentioned the pros and cons of several cache replacement mechanisms in the related work section. Our proposed mechanism uses a unique content popularity mechanism to replace the contents in the cache memory. Initially, the popularity of a content (P_C) is zero. If any content request packet is served from a node, then a small impact, α, is added with P_C and; therefore, the increased popularity of that content becomes $P_C = P_C + \alpha$. The value of α is 0.01 and it is continuously increased after a successful content hit in the cache memory of the node. Similarly, we calculate the content popularity for each content in the cache memory of a node. The contents are cached into the cache memory of a node with the highest P_C values. If the cache memory size of node n is $£_n$, the existing cached content size of a node is λ, incoming content size is β with content popularity, P_{Ci}, then the Algorithm 2 is used to take the decision of content caching.

Algorithm 2: An algorithm for content replacement in the cache memory.

Input: β, P_{Ci}

Output: Whether the cache memory is updated or not.

```
Definition of used variables:
£n = The cache memory size of node n
λ = Existing cached content size of a node
β = Incoming content size
PCi = Incoming content popularity
PCl = The lowest content popularity among the cached content
Initialization:
For each content in the network PC = 0;

Begin;
  Calculate the free cache memory (FCM) = £n − λ
    if FCM ≥ β
      Cache the incoming content
    else if FCM < β
        if PCl > PCi
          Do not cache the incoming content
        else if PCl ≤ PCi
            Find the lowest content with PC
            Select those contents where the summation of content size with lowest PC ≥ β
            Replace those contents by incoming contents
  End;
  return output;
```

4.1.2. Responsibility of the Clustered Node and Cluster Head Node

The cluster nodes are the edge device of the access network in some cases. Therefore, they are receiving the content request message directly from the access network devices. They are located in one-hop or two-hop distance of content requester and the other access network devices are also available in the one-hop distance of content requester. When a cluster node receives any content request message from the content requester, the node itself tries to serve the request if the requested content is available in the CS of the node. If the content is not available in the CS, then the content request message is forwarded to the cluster head and the cluster head tries to serve the request based on the availability of the content. If the content is not available in the cluster head, the cluster head checks its repository to find the availability of the content within the cluster because the cluster head knows the available content information within the cluster. We apply algorithm 3 to make a cluster head from a cluster.

Algorithm 3: An algorithm to make a cluster head.

Input: *C_List*

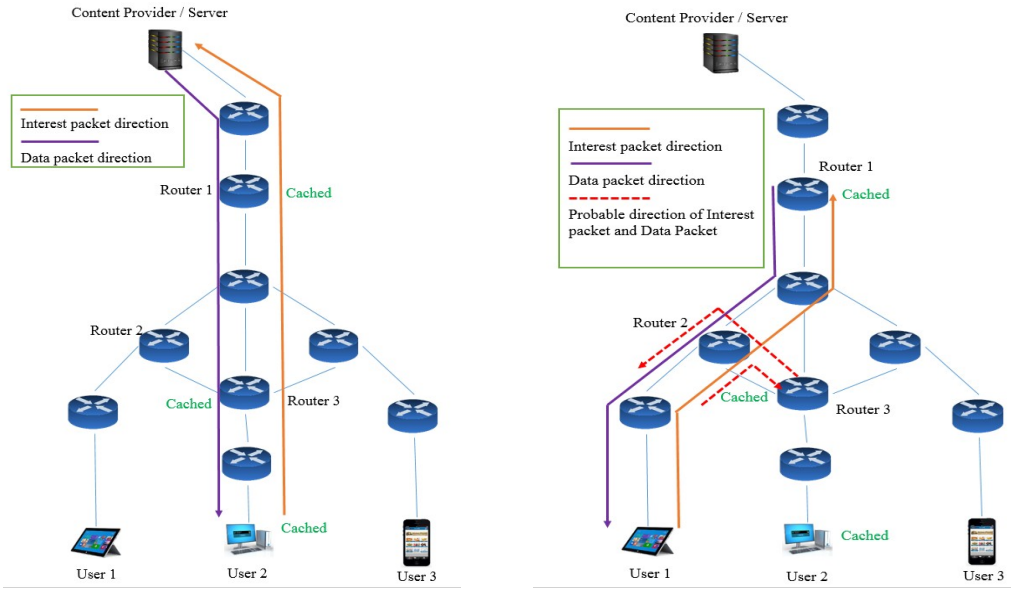
Output: Cluster_head_list, *k*th list

```
Definition of used variables:
MAX_mem = Maximum memory of a node;
Cluster_head_list = List of the heads of all clusters;
C_List = List of all clusters in the network;

Initialization:
For each Cluster k = 1 to Cluster_head_list ≠ Ø;

Begin;
 For each node j = 1 to n, in cluster k;
    Inform self_memory to all cluster members;
    Self_memory = node[j] memory;
        For each node Do
            if Self_memory > Neighbor(j) memory
                MAX_mem = Self_memory;
            else if Self_memory == MAX_mem
                Cluster_head = node j in Cluster k;
      Cluster_head_list = Cluster_head_list ∪ j;
      kth list = jth node is the head of the kth cluster;
  k++;
  Continue until C_List ≠ Ø;
 End;
 return Cluster_head_list and kth list;
```

The content request message is forwarded to the neighbor cluster head if the content is not available within the cluster. Similarly, the neighboring cluster heads do the same procedure to serve the request. Following these procedures, a content request message is served by the core network. The cluster head handles the caching mechanism within the cluster by keeping the information of all the cluster nodes. The most popular content is cached in the cluster head within the core network. Each node knows the available free cache memory of self-node and they share this information with the cluster head. As a result, the cluster head can share the more popular content with them, and the popular or the less popular contents are also cached based on the availability of the free memory in the cluster node. The cluster head also shares the information with the neighboring cluster head and maintains the same caching mechanism.

4.1.3. Enhancement of Existing Caching Mechanisms Based on the Clustering Approach

The on-path and hash caching problems of in-network caching mechanisms are described in Section 3. The solution to the on-path and hash caching mechanisms is described in this section based on the clustering mechanism in CCN, a type of ICN. The following Figure 5 illustrates a specific on-path caching problem that occurs frequently in the large network scenario. A similar type of scenarios is available in the core network that was presented in Figure 4. We consider CCN, an ICN type to explain accurately based on the CCN communication mechanism. We can consider the Tier 4 devices as content provider, centralized cloud server, or service provider. The users are in Tier 1, and they are directly connected to the routers. We also consider cellular network and Packet Data Network Gateway (PGW) in Tier 3, that are collocated with the router, as the devices of the core network. In the LTE network, several eNBs are connected to one PGW. Therefore, the most popular contents are cached in the PGW if no types of EC are applied in the eNBs.



(a) Interest and Data packet flow (b) Probable Interest and Data packet flow

Figure 5. The problem with on-path caching.

The problem of the on-path caching mechanism is shown in Figure 5a,b. Figure 5a shows that User 2 sends an interest packet indicated by an orange line to retrieve the content from the content provider. The requested interest packet is served by the content provider and the content is forwarded to the content requester following the reverse path of the interest packet that is indicated by the purple line in Figure 5a. The incoming data packet is cached in Router 1 and Router 3 according to the on-path caching mechanism. However, when User 1 sends another interest packet for the same content, the interest packet is forwarded to Router 1 instead of Router 3 according to the rules of the on-path

caching mechanism. Figure 5b shows the probable forwarding direction of interest packet indicated by the red line. This is a concerned forwarding issue of the on-path caching mechanism that can be solved by our cluster-based mechanism. The content is efficiently cached in the core network devices or the edge devices of core and access networks based on content popularity. Therefore, the network ensures the end-users for accessing the contents from the nearby cached devices. If the content is the most popular in the network, then the end-users can access directly from the edge-devices of the network.

We solved the problem of the on-path and hash caching problem by our cluster-based mechanism where we applied the Algorithm 1 to make the cluster from the core network devices and Algorithm 3 to decide the cluster head for each cluster. Figure 6 shows the solved on-path caching mechanism using a clustering approach. The circle in Figure 6 indicates the cluster where User 1 and User 2 are connected. When User 1 sends the interest packet to the connected Router, Router forwards the interest packet to Router 3 according to the interest packet forwarding strategies in the cluster-based approach. Router 3 replied with the content in the reverse path of the forwarded interest packet. The green line in Figure 6 shows the direction of interest and data packets respectively for User 1. As a result, the content transfer time is decreased, the consumed network bandwidth is decreased, and the overall network performance is increased.

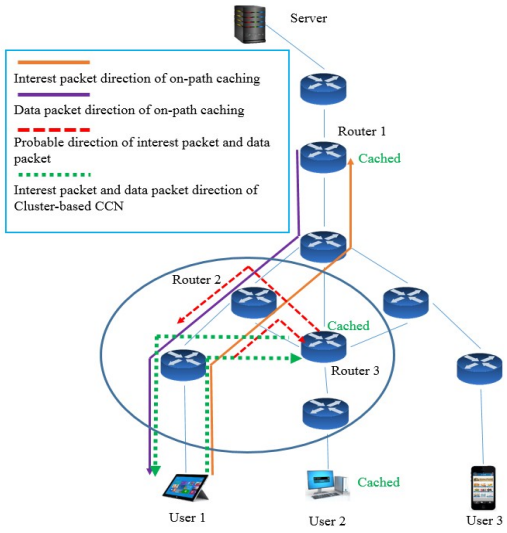


Figure 6. Enhanced on-path caching.

Another caching mechanism called hash caching [31] was proposed and Figure 7 shows the procedure of the hash caching mechanism. According to the rules of the hash caching mechanism, User's interest packet is forwarded to the Router 2 although the content is available in the nearby content server. As a result, the network overhead is increased as well as the content transmission time. This drawback of the hash caching mechanism can be easily solved by the cluster-based caching mechanism approach.

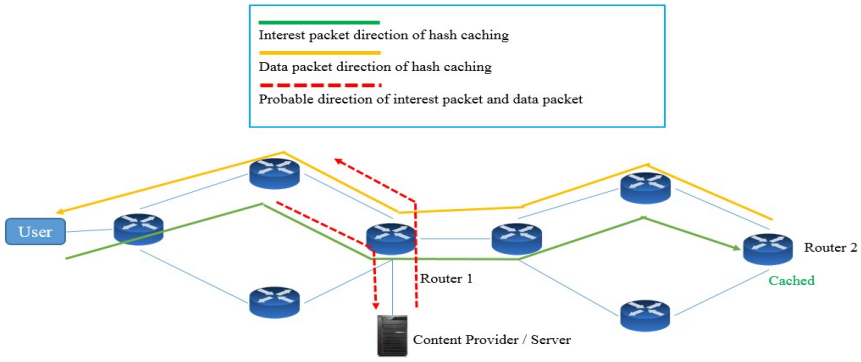


Figure 7. Problem with hash caching.

The solution of the hash caching problem is provided based on the cluster-based approach that is shown in Figure 8. At first, the clustering approach applied to make the cluster and cluster head. When the neighbor router of the User is unable to serve the interest because of the unavailability of the content, the interest packet is forwarded to the Router 4 according to the interest packet forwarding mechanism in the cluster-based mechanism. Router 4 forwards the interest packet towards the content server instead of Router 2. Finally, the content server serves the interest packet to the User.

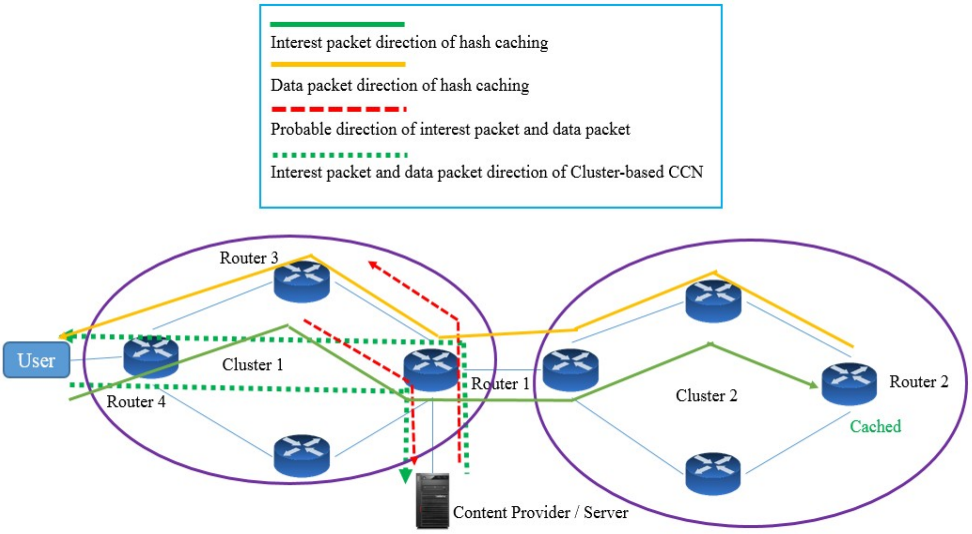


Figure 8. Enhanced hash caching.

4.1.4. Content Access Mechanism

The contents are accessed from Tier 1 devices. We classified the end-users into two categories: (1) Static or mobile end-users, and (2) end-users based on the content generator and consumer as we discussed before. Most of the IoT and IIoT devices are the content generators. The content consumers generate the content request packet and send it to the nearby connected devices. To describe the detailed content access mechanism, let us consider the Table 1 variable sets for including the definitions and notations. We use a set of EC types, ς, in Tier 2 to provide the benefits of cloud computing and caching in the edge on the core and access network devices. All devices have the caching capabilities in Tier 2 and only EC has the computing capabilities. The combined caching resources of Tier 2 is CRL2. The list of the eNBs of Tier 2 is defined as L_{eNB}. The Tier 2 devices are the neighbor device of Tier 1 devices. The velocity of Tier 1 mobile devices is V_m. The tier devices request ξ types of contents. The components of each content are defined as $\mu_i = \{\beta, P_C, t_H, C_R\}$, where $i \in \xi$, β is the size of the content, P_C defines the content popularity, t_H is the last content hit or access time, and C_R indicates any types of computing requirements for the content.

An end-user sends a content request message for a type of content that is available in Tier 4, Tier 3, Tier 2, or the neighboring devices. The underlying communication technologies help the end-user to communicate with other devices. Generally, the Tier devices are directly connected to the Tier 2 devices. Therefore, the content request message is forwarded to Tier 2 devices. The content request message may be served by any Tier devices and the content will be forwarded to the content requester following the reverse path of the content request message. The served content is cached based on the content popularity mechanism that was described in Algorithm 2. There are enough caching resources in l_2, and l_2 is located at one-hop distance from the end-users. Therefore, we optimize the use of caching resources of corresponding l_2 devices. Let $P_{i,j}$ denotes the allocated caching resource, j on corresponding l_2 devices for incoming content i, where $i \in \xi$, $j \in l_2$, and $\varsigma \in l_2$. Considering the storage limitations in the l_2 devices except for ς, the caching resources of ς cover the storage limitations. The computing resources of ς, $\Psi_{i,j}$ is allocated to the computing resource, j on the corresponding ς for incoming content i, where $i \in \xi$, $j \in \varsigma$.

Table 1. Used notations.

Definitions	Notations
Set of the EC types (MEC, MCC, FC, Cloudlets)	ς
Set of Tier 2 devices	l_2 = {BSs, APs, ς, … }
Set of N eNBs	δ_{eNB}
The covered area of a mobile device in Tier 2	Γ_m
Set of *N* Aps	$¥_{AP}$
Caching resources in Tier 2	CRL_2
Caching resources in one eNB	C_{eNB}
Caching resources in one AP	C_{AP}
Caching resources in one ς	C_ς
Computing resources in one ς	CO_ς
End-user types based on the position	EU_{tp}
End-users based on the content generator or consumer	EU_{gc}
Speed of the end-user mobile device	V_m
List of the eNBs, (eNB_1, eNB_2, … … . eNB_N)	L_{eNB}
Types of the requested content	ξ
The component of each data type	μ_i = {β, P_C, t_H, C_R}
Caching resource of l_2, j used to process content i	$P_{i,j}$
Computing resource of ξ, j used to process content i	$\Psi_{i,j}$
Getting a unit content from mobility supported content provider in Tier 2 devices	τ_m
Getting a unit content time from Tier 4 devices	τ_{cs}
Getting a unit content time from a cached content device	τ_{cn}
Processing latency for C_R	τ_R
Generated data by an IoT device	G_{IoT}

To minimize the average latency [37] of the contents downloading process, we formulate our efficient caching mechanism based on Equation (1).

$$\min_{\{P_{i,j},\, \psi_{i,j}\}} \sum_{i=1}^{\xi} P_{c_i} \sum_{j=1}^{J_i^{max}} \frac{\left\{\left(\frac{\Gamma_m}{V_m \tau_m} - P_{i,j}\right)\tau_{cs} + \psi_{i,j}\tau_R + P_{i,j}\tau_m\right\}}{\xi}, \tag{1}$$

where τ_m, τ_R, and τ_{cs} definitions are defined in Table 1. The distance between the Tier 4 devices and Tier 1 devices are long. Therefore, $\tau_{cs} > \tau_m$ and the required computing time, τ_R, depends on the types of computing algorithm and the size of the content. If mobility occurs in the mobility supported devices in Tier 2 during the content access, then the contents are cached into multiple mobility supported

devices (e.g., eNBs and APs). So, j_i^{max} is the index of the possible BS where type i content is cached in and j_i^{max} can be defined by Equation (2).

$$\sum_{j=1}^{J_i^{max}-1} \frac{\Gamma_m}{V_m \tau_m} < \beta_i \leq \sum_{j=1}^{J_i^{max}} \frac{\Gamma_m}{V_m \tau_m}. \tag{2}$$

There are several resource constraints (RCs) during caching and computing in the respective devices. Equations (3) and (4) show the RC of allocated caching resources and computing resources among the storage capacity of Tier 2 devices. Equation (5) shows the RC of the size of cached content in mobility-supported content providers in Tier 2 devices, where the cached content size should not exceed the maximum amount that is required for the end-users. Equation (1) and the three RCs are applicable for Tier 2 devices to support the mobility of the end-users.

$$RC1: \quad \sum_{i=1}^{\xi} \frac{P_{i,j}}{1 + C_R \, \psi_{i,j}} \leq CRL_2, \; j \in CRL_2 \; and \; \forall (C_{eNB}, \; C_{AP}, \; C_\varsigma) \in CRL_2. \tag{3}$$

$$RC2: \quad \sum_{i=1}^{\xi} \psi_{i,j} \leq CO_\varsigma, \; and \; j \in \varsigma. \tag{4}$$

$$RC3: \quad P_{i,j} \leq \frac{\Gamma_m}{V_m \tau_m}, \; i \in \xi, \; and \; j \in CRL_2 \; and \; \forall (C_{eNB}, \; C_{AP}, \; C_\varsigma) \in CRL_2. \tag{5}$$

If the Tier 2 devices do not support mobility, then Equation (1) is modified into Equation (6) where $\tau_m = 0$, $\Gamma_m = 0$, and $V_m = 0$. Similarly, Tier 3 and Tier 4 devices also do not support mobility. There are some content providers that also do not support mobility. Therefore, when the end-users are accessing content from any devices where the content is already cached and do not support mobility, then Equation (7) is applied and the value τ_m is replaced by τ_{cn}.

$$\min_{\{P_{i,j}, \psi_{i,j}\}} \sum_{i=1}^{\xi} P_c \sum_{j=1}^{J_i^{max}} \frac{\left(\psi_{i,j} \tau_R + P_{i,j} \tau_m\right)}{\xi}. \tag{6}$$

$$\min_{\{P_{i,j}, \psi_{i,j}\}} \sum_{i=1}^{\xi} P_c \sum_{j=1}^{J_i^{max}} \frac{\left\{\left(\frac{\Gamma_m}{V_m \tau_m} - P_{i,j}\right)\tau_{cs} + \psi_{i,j} \tau_R + P_{i,j} \tau_{cn}\right\}}{\xi}. \tag{7}$$

If the cached content devices do not support mobility, then $\tau_m = 0$, $\Gamma_m = 0$, and $V_m = 0$. Therefore, we can rewrite the Equation (7) as Equation (8).

$$\min_{\{P_{i,j}, \psi_{i,j}\}} \sum_{i=1}^{\xi} P_c \sum_{j=1}^{J_i^{max}} \frac{\left(\psi_{i,j} \tau_R + P_{i,j} \tau_{cn}\right)}{\xi}. \tag{8}$$

The EU_{tp} devices are accessing the contents from different tiers based on their needs. The contents are cached into several devices in different Tiers based on the content popularity and other parameters that were described before. We described the procedure for efficient use of caching and computing resources in the static and mobile environment. The cellular network mainly provides the mobility support and the caching strategy also described during mobile and static environments for end-users. The coexistence of ICN and EC mechanisms provides us both caching and computing resources and it increases the overall performances of the network.

4.2. Upstream Data Handling

A lot of IoT and IIoT devices generate a huge amount of variable size data packets periodically. The generated data are used locally and then these generated data stored for future data analytics or other purposes. We use four different components for each data type. The generated data is forwarded to the l_2 devices based on the connection and network type between Tier 1 and Tier 2 devices. ς mechanisms are used to the l_2 devices and the ς mechanism provides caching and computing resources to the l_2 devices. The computing resources provide the end-users with the expected computing environments (e.g., task offloading to the ς mechanisms to reduce their workload). Similarly, the video analytics capability, big data analysis based on the end-user expectation, and location-oriented services are also provided by the ς mechanisms. After receiving the contents from the end-users, ς mechanisms apply the C_R computing mechanism to the collected data based on the end-user expectation. To apply the C_R computing mechanism to the collected data, ς mechanisms need τ_R processing time.

$$\min_{\{P_{i,j},\psi_{i,j}\}} \left\{ \sum_{j=1}^{N} C_{R_j} \left(\sum_{i=1}^{N} M_i \sum_{t=t_1}^{t_2} G_{IoTt} \right) \right\}. \tag{9}$$

Let us consider a starting time t_1 and ending time t_2 for collecting the data that is generated from a single IoT device within Δt time. There is a total number of M IoT devices in the network. Their generated data are sent to the Tier 2 and ς mechanisms stored using their cache resources. Then, ς mechanisms apply C_R different types of computing algorithms to the collected data. Therefore, the total number of caching and computing resources are calculated based on Equation (9). Finally, the processed combined data are stored in the local cloud data server or forwarded to the centralized cloud server based on the user expectation. As a result, the volume of the data is reduced and the data traffic in the network is reduced; content transfer time is also reduced. Moreover, the overall performance of the network is increased.

4.3. Future Work Directions

We provided an in-depth analysis of the coexistence of the types of EC and the ICN concept for downstream and upstream content flow handling. This paper described the detailed mechanism for downstream content flow based on the cluster-based approach in Tier 3 of our proposed architecture. The upstream data handling includes big data management issues, data analytics, computing-related issues, video analytics, and many other data-driven applications. These topics are suitable to handle by using our proposed Tier-based network architecture. We can apply MCC, MEC, FC, and Cloudlets concepts in Tier 2 based on the application types or research area. In the V2X communication and autonomous driving scenarios, a lot of computations are required. Therefore, MEC can be used to provide the location-oriented services and computing aspects for autonomous driving vehicles. Similarly, FC or Cloudlets can be used in the IoT or IIoT environments to offload a task or caching purposes and MCC can be used for big data analysis. Similar architecture can be used by adopting the necessary types of EC mechanism. Therefore, we will get all the benefits of the proposed architecture. This architecture opens a new research dimension for the data-driven IoT applications and the multimedia content delivery with low latency in ICN.

5. Performance Analysis

We considered several topologies to evaluate the performances of our proposed architectures for caching mechanisms and our main topologies contain 35 nodes. These nodes represent the content providers, routers, end-users, and other proximity devices, and their identification starts from node number zero to thirty-four. In our topological environment, each node has all the capabilities like a CCN-enabled device can do. We enabled all the nodes as a CCN node and they can generate interest packets, they can work as a router, and they can work as a content provider. These nodes are considered

as a 7 × 5 matrix in a two-dimensional space. The distance between two nodes is 40 m and a node is connected to all the neighbor nodes. Each row has five nodes and is considered as the one hop of the devices and there are total seven hops in our topologies. The node zero to four are located at a six-hop distance from the end-users. Similarly, the other nodes are located to the other hops consecutively from the end-user devices. Any node of any hop can work as a content provider or consumer or router. Following these characteristics, we had simulated several scenarios where the content providers' and consumers' distance is at about five hops. We will describe the simulated results based on the scenarios that we have used for simulations.

We simulated and evaluated our proposed mechanisms in the Network Simulator-3 (NS-3) environment. Direct code execution (DCE) [38] was used to apply the CCNx application [38] in the NS3 environment. The CCNx application was used to apply the CCNx functionalities at each device in the simulation environment and the different modules of the CCNx application were used in the C++ environment to apply the clustering concept in the simulated network environment. Table 2 parameters were used during the simulation.

Table 2. Simulation parameters.

Name of the Parameters	Value
Network type	Wired, wireless
Simulation time	5~20 min
Underlying transport layer	Transmission Control Protocol (TCP)
Bandwidth in wired line	1 Gbps
Bandwidth in wireless line	1, 2, 5.5, 11 Mbps
eNB's power	46 dBm
Delay	2 ms
Packet size	8.2, 8 KB
File size	8.2 KB, 1 MB, 4.1 MB

5.1. Performance Analysis of Caching Mechanisms

The state-of-the-art mechanisms cache the content in several inefficient ways (e.g., LCE caches the content to all the traversed network devices or LCD caches the content only in the traversed neighbor node). Therefore, the duplicate content increases the memory consumption, but it also increases the content hit ratio in case of LCE. There are several inefficient cache replacement algorithms (e.g., least recently used (LRU), first in first out (FIFO), and last in first out (LIFO)). As a result, the probability of content replacement from the cache memory is very high in the dense network area. These cache replacement algorithms are directly related to the content hit ratio that was used in several state-of-the-art mechanisms. Therefore, the probability of replacement of the most popular content was increased. However, our proposed mechanism provides better caching efficiencies in terms of caching popular content. The detailed description of the proposed solutions based on the cluster-based mechanism was already shown in Figures 6 and 8. In Figure 9, we also showed the performances of our proposed, on-path and the state-of-the-art caching mechanisms in terms of content transfer time. A small video file was used as video traffic to evaluate the content delivery time for all mechanisms. It is clear from Figure 9 that our cluster-based approach required less content transfer time than state-of-the-art mechanism if the content is not available in the cache memory in case of the basic LCE. The content transfer time in the LCE mechanism is equal to our cluster-based CCN but the memory consumption, bandwidth consumption, network congestion, and flooding the network reduces the overall network efficiencies. Therefore, the cluster-based CCN provides a better solution than the existing simulated state-of-the-art mechanisms.

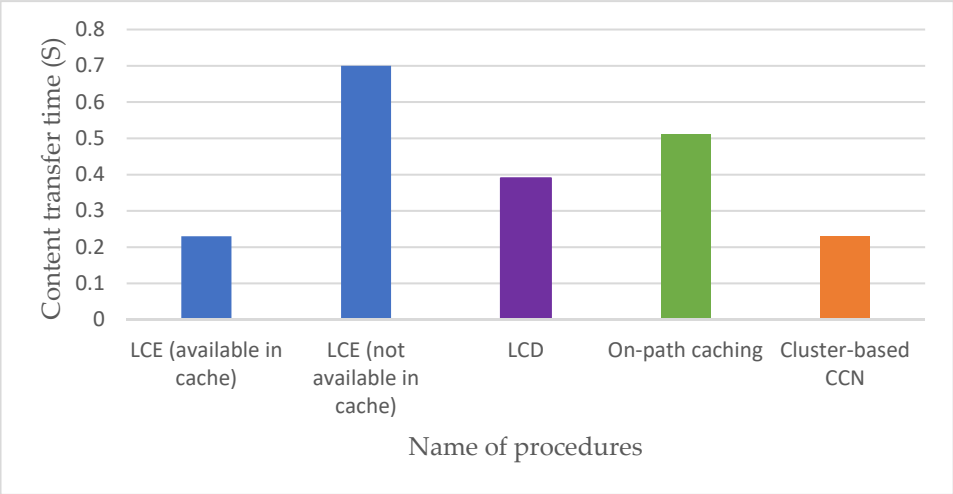


Figure 9. Comparison among proposed and state-of-the-art caching mechanisms.

The performances of the hash caching, proposed cluster-based and state-of-the-art caching mechanisms are shown in Figure 10 in terms of content delivery time. We already described the problem and the solution of hash caching in Figures 7 and 8. We considered the same simulation scenario for all the caching mechanisms. The simulation result shows that our proposed cluster-based mechanism needs less time than the hash caching mechanism but similar to the state-of-the-art caching mechanisms. Although the state-of-the-art caching mechanisms need equal time compared to our proposed mechanism, the drawbacks of LCE and LCD do not motivate to be used in the real network environment. The cluster-based approach provides better results in hash caching and on-path caching mechanism due to our proposed efficient cache replacement mechanism in the cluster-based network environment.

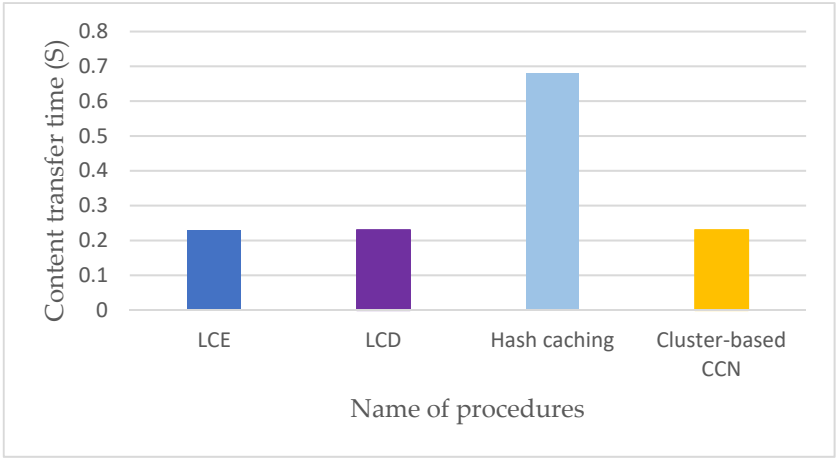


Figure 10. Comparison among proposed and state-of-the-art caching mechanisms.

5.2. Impact of Hop-Based Caching

The desired contents for the end-users may be situated in the one-hop, two-hop, or multi-hop distances from the end-users or the content requesters. If several hops are required to reach the content providers, how it may affect the network is very important because forwarding an interest packets towards one more hop means the consumption of network bandwidth, increase in network traffic, the potential creation of congestion, and the potential increase of the packet loss ratio. Therefore, serving the interest packet from nearby content providers may save network bandwidth, reduce the network traffic in a certain region, reduce the network congestion, and reduce the packet loss

ratio. All these parameters are directly related to the content access time. Therefore, we simulated a network topology where the same content was available at one-hop to six-hop distances from the content consumers. The content consumer was in the last row of the matrix and the node number was thirty-two. Figure 11 shows the impact of hop-based caching where a content consumer was sending the interest packet for the same contents where the content providers were in one-hop to six-hop distances from the content consumers. Although the simulation result shows that the content access time is reducing, at the same time the other parameters are also getting the benefits if the content access time is low. If the required contents can be served from one-hop or two-hop distances, then the bandwidth will be saved above two-hop distances from the end-users. Similarly, the network traffic will reduce, and network congestion will be affected above the two-hop distance.

5.3. Impact of Content Popularity in Caching

The end-users access the popular contents frequently. If a popular content is accessed by an end-user at any place, it inherits the value of the content popularity. Therefore, the popular content is easily replaced by a non-popular content at the edge of the core and the access network devices. Moreover, the popularity of the content increases continuously based on every cache hit. We considered the same topology and simulated the same topology by adding two more content consumers. They were the proximity devices of each content consumer. The newly added content consumer node numbers were thirty-three and thirty-four. The node number thirty-two was requesting different contents from the six different content providers that were located at the one-hop to six-hop distances from the requested content consumers. The node number thirty-three is the neighbor node of thirty-two, also requesting for the same content after a while. Similarly, the node thirty-four was also requesting for the same content. These three nodes generated eighteen interest packets for the six different contents those are available in one-hop to six-hop distances from the content requesters. We provided a comparative analysis among the current Internet architecture, basic CCN architecture, and our proposed cluster-based CCN architecture.

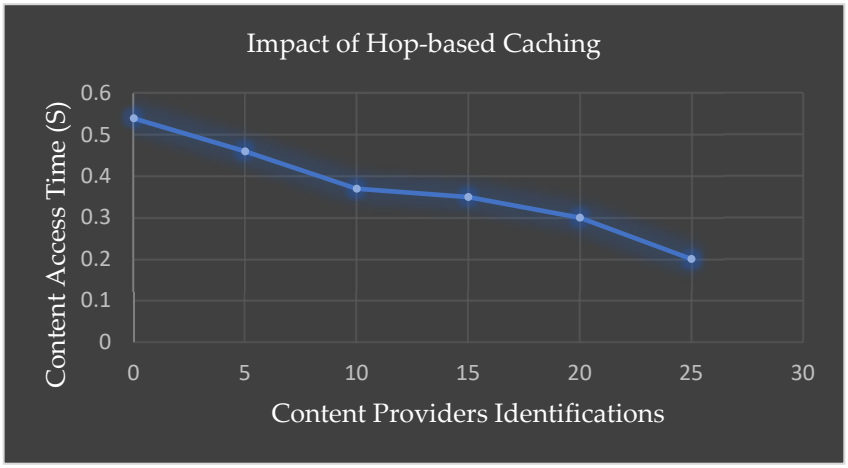


Figure 11. Impact of hop-based caching.

The current IP-based Internet architecture forwarded these eighteen interest packets towards the content provider although, there were only six distinguishable interest packets forwarded towards the network. Therefore, the network congestion may occur, consumption of network bandwidth increases, and the network traffic increases in the network. Although, the basic CCN forwarded only six interest packets, they flooded the network. As a result, similar problems may occur. We provided an intensive cluster-based effective solution that solved all the above-mentioned issues. The cluster-based approach does not flood the network and maintain an efficient caching mechanism based on content popularity. Therefore, the content access time was reduced for similar interest packet requested from

the nearby devices. Figure 12 shows the simulation result, which displays how the efficient caching mechanism affects the content access time. Node thirty-two was accessing the contents from different content providers but node thirty-three was accessing the contents from node thirty-two, and node thirty-four was accessing the contents from node thirty-three. Therefore, Tier 2, 3, and 4 devices were free from all kinds of network traffic and bandwidth consumptions due to the content accessing from the nearby cached memory. Our cluster-based network always ensured that the content is accessible from the nearby devices and manage the cache memory of each node efficiently based on our proposed popularity-driven content replacement mechanism. Our mechanism also ensures low latency communication in every scenario of downstream content flow.

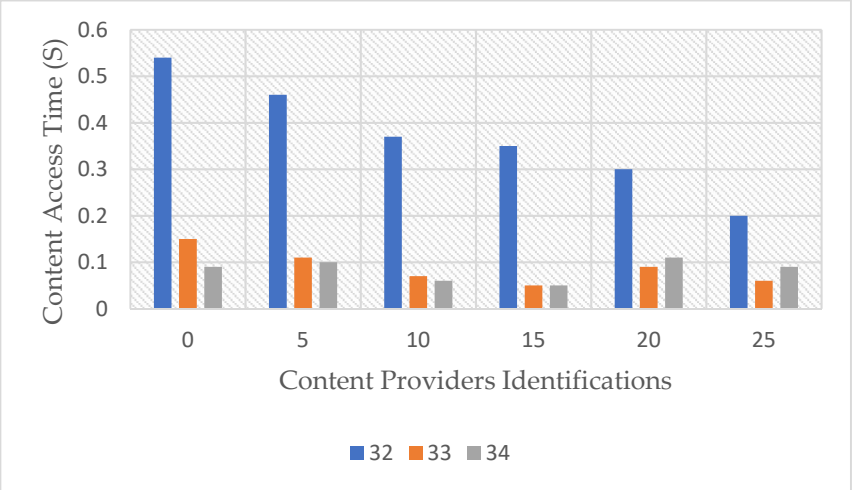


Figure 12. Impact of popular content in caching.

5.4. Impact of Duplicate Content in Caching

The contents were cached in the several nodes in the network, although we optimized the duplicate content caching based on the clustered method and the cluster head maintains the duplicate content to the proximity devices in the network. The cache memory was reused for duplicate content, but it does not affect the content access time due to duplicate content. We simulated a network topology where content was available in the four-hop, five-hop, and six-hop distances from the content requesters. Nodes zero, five, and nine also contained the same content, but they were located at one-hop distance from each other. Our main objective is to show that there is no impact of duplicate content in our cluster-based content access mechanism. Our mechanism always ensured that the nearby content providers served the contents to the end-users. Figure 13 shows the simulation result where maximum available contents were accessed within a shorter time from the nearby content providers, although the same content was available in the remote content providers.

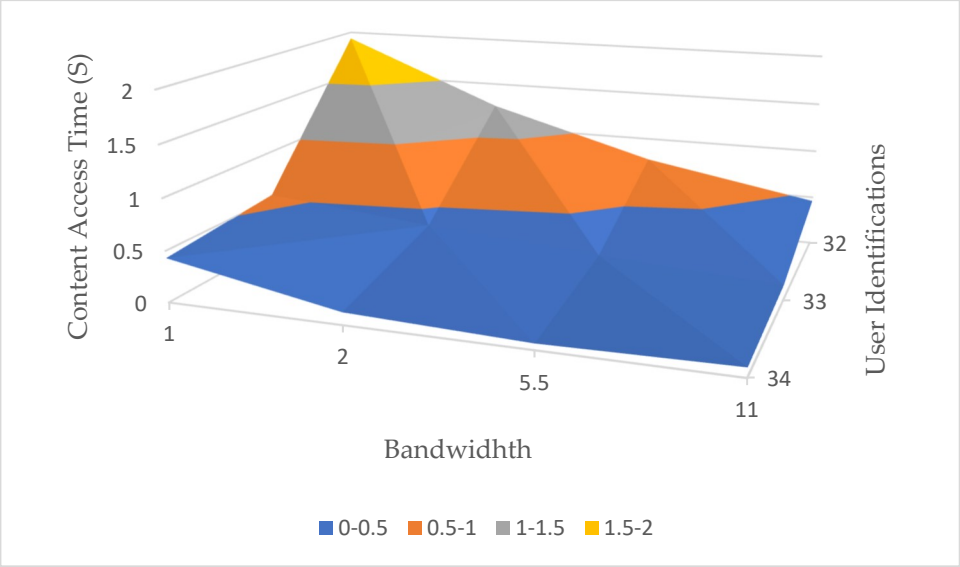


Figure 13. Impact of duplicate content in caching.

5.5. Performance of Caching in the LTE Network

In our proposed architecture, we showed the cellular network (i.e., LTE network) whereby the content providers were in the Tier 4 and the PGW is situated in Tier 3. The eNBs are in Tier 2 and the end-users of the LTE network were available in Tier 1 devices. The end-users had several network interfaces (e.g., cellular, Wi-Fi, etc.); those were used to access the contents in different scenarios. There were several transmission modes in the LTE network. From those LTE transmission modes, we used two transmission modes in our simulation environments. One was single input single output (SISO) and the other one was multiple input multiple output (MIMO). Figure 14 shows the simulation result for the LTE network. The simulation result shows that MIMO provides almost double throughput compared to SISO at 10 MHz channel bandwidth, but our main objective is to show the performance of caching in different Tiers. Although the simulation result is for the same transmission mode, the content access time is less based on the location of the content providers. In the LTE network, multiple eNBs were connected to the PGW that was in Tier 3. The rest of the core network and Tier 4 devices were connected to the LTE network through the PGW. Therefore, PGW plays an important role in the LTE network to reduce the content access time and to increase the probability of content access from nearby devices. Moreover, the PGW was the part of the clustered devices and it could be the cluster head of a cluster. As a result, the direct communication among the cluster head also increases the probability of content access from nearby devices where the contents are cached.

The types of EC were the most important factor for the cellular network because it has caching and computing capabilities including other cloud-related characteristics. We did not show the performances of the combined situation of the eNB's and the types of EC mechanisms. But, surely, the presence of any types of EC increases the caching and computing capabilities in the edges of the core and access network devices. Therefore, the probability of the more popular content caching increases within the one-hop distance of the end-users. Therefore, the end-users can directly access the most popular content from the one-hop distance and the content access time reduces a lot. The coexistence of EC and ICN concepts also ensures the low latency communication in the cellular network.

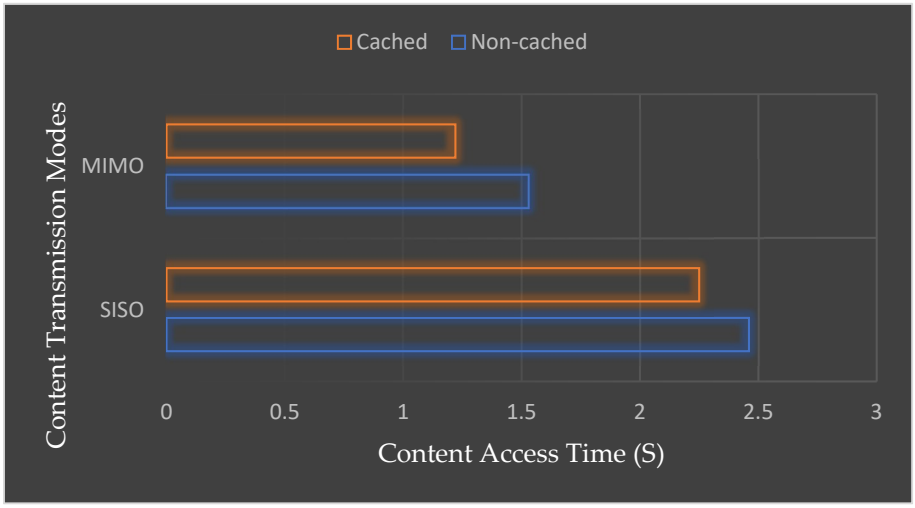


Figure 14. Performance of caching in the LTE network.

6. Conclusions

In this paper, we proposed an architecture based on the coexistence of ICN and the types of EC to provide an efficient caching mechanism for the data-driven IoT applications and to deliver multimedia information with low latency in ICN. We applied the clustering mechanism in our proposed tier-based architecture. We also proposed a new mechanism to identify the popular content in the network and the content replacements were done based on the newly-proposed content popularity mechanism. The drawbacks of the existing on-path and the hash caching mechanisms were described in this paper. We showed that our proposed cluster-based approach required less content transfer time than existing caching schemes via simulation and mathematical analysis and the overall performance was better than other caching mechanisms in terms of content transfer time and efficient content caching. We also showed that our cluster-based approach solved the existing drawbacks of on-path and hash caching mechanisms. Our cluster-based mechanism ensured fast content retrieval from the nearby content provider to the end-users.

We simulated several wireless scenarios including the LTE network to ensure the efficiencies of our proposed cluster-based efficient caching mechanism. We showed the performance results depending on the locations of the content providers in a hop-based manner, the impact of popular content caching in the edge of the core and access network devices, and how duplicate content changes the content access time. In every scenario, we confirmed that the content is accessed from the nearby content providers or popular cached devices. Therefore, the content access time is reduced and the probability of the content access from the nearby devices is increased. Moreover, the network traffic is reduced, the consumption of the network bandwidth is reduced, network congestion is reduced, the packet loss is also reduced, and the overall performance of the network is increased.

Author Contributions: Conceptualization, K.H. and S.-H.J.; methodology, K.H.; software, K.H..; validation, K.H. and S.-H.J.; formal analysis, K.H. and S.-H.J.; investigation, K.H. and S.-H.J.; writing—original draft preparation, K.H.; writing—review and editing, K.H. and S.-H.J.; supervision, S.-H.J.; project administration, S.-H.J.; funding acquisition, S.-H.J.

Funding: This research was financially supported by the Ministry of Trade, Industry and Energy (MOTIE) and Korea Institute for Advancement of Technology (KIAT) through the International Cooperative R&D program. This work was supported by Hankuk University of Foreign Studies Research Fund of 2019.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AR	Augmented Reality
BAN	Body Area Network
CC	Cloud Computing
CS	Content Store
CCN	Content-Centric Networking
DCE	Direct Code Execution
EC	Edge Computing
ETSI	European Telecommunications Standards Institute
FC	Fog Computing
FIFO	First in First out
FIB	Forwarding Information Base
ICN	Information-Centric Networking
IoT	Internet of Things
IIoT	Industrial Internet of Things
IP	Internet Protocol
LCD	Leave Copy Down
LIFO	Last in First Out
LRU	Least Recently Used
LCE	Leave Copy Everywhere
MCC	Mobile Cloud Computing
MEC	Mobile Edge Computing
MIMO	Multiple Input Multiple Output
NDN	Named Data Networking
NS3	Network Simulator-3
PIT	Pending Interest Table
PGW	Packet Data Network Gateway
QoS	Quality of Services
SISO	Single Input Single Output
V2X	Vehicle to Everything
VR	Virtual Reality
VM	Virtual Machine

References

1. Guo, X.; Chu, L.; Sun, X. Accurate Localization of Multiple Sources Using Semidefinite Programming Based on Incomplete Range Matrix. *IEEE Sens. J.* **2016**, *16*, 5319–5324. [CrossRef]
2. Wang, L.; Ranjan, R. Processing Distributed Internet of Things Data in Clouds. *IEEE Cloud Comput.* **2015**, *2*, 76–80. [CrossRef]
3. Quwaider, M.; Jararweh, Y. Cloudlet-based efficient data collection in wireless body area networks. *Simul. Model. Pract. Theory* **2015**, *50*, 57–71. [CrossRef]
4. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge Computing: Vision and Challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [CrossRef]
5. Satyanarayanan, M.; Bahl, P.; Caceres, R.; Davies, N. The Case for VM-Based Cloudlets in Mobile Computing. *IEEE Pervasive Comput.* **2009**, *8*, 14–23. [CrossRef]
6. Xylomenos, G.; Ververidis, C.N.; Siris, V.A.; Fotiou, N.; Tsilopoulos, C.; Vasilakos, X.; Katsaros, K.V.; Polyzos, G.C. A Survey of Information-Centric Networking Research. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1024–1049. [CrossRef]
7. Li, G.; Li, J.; Wu, J. Fog-enabled Edge Learning for Cognitive Content-Centric Networking in 5G. *arXiv* **2018**, arXiv:1808.09141v1.
8. Lei, L.; Xiong, X.; Hou, L.; Zheng, K. Collaborative Edge Caching through Service Function Chaining: Architecture and Challenges. *IEEE Wirel. Commun.* **2018**, *25*, 94–102. [CrossRef]

9. Thomas, D. Cloud Computing—Benefits and Challenges. *J. Object Technol.* **2009**, *8*, 37–41. [CrossRef]
10. Khan, A.U.R.; Othman, M.; Madani, S.A.; Khan, S.U. A Survey of Mobile Cloud Computing Application Models. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 393–413. [CrossRef]
11. Rahimi, M.R.; Ren, J.; Liu, C.H.; Vasilakos, A.V.; Venkatasubramanian, N. Mobile cloud computing: A survey, state of art and future directions. *Mob. Netw. Appl.* **2014**, *19*, 133–143. [CrossRef]
12. Shaukat, U.; Ahmed, E.; Anwar, Z.; Xia, F. Cloudlet deployment in local wireless networks: Motivation, architectures, applications, and open challenges. *J. Netw. Comput. Appl.* **2016**, *62*, 18–40. [CrossRef]
13. Hu, P.; Ning, H.; Qiu, T.; Zhang, Y.; Luo, X. Fog Computing Based Face Identification and Resolution Scheme in Internet of Things. *IEEE Trans. Ind. Inform.* **2017**, *13*, 1910–1920. [CrossRef]
14. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the Internet of Things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; ACM: New York, NY, USA, 2012; pp. 13–16.
15. European Telecommunications Standards Institute Industry Specifications Group, Mobile-Edge Computing Service Scenarios. Available online: http://www.etsi.org/deliver/etsi_gs/MEC-IEG/001_099/004/01.01.01_60/gs_MEC-IEG004v010101p.pdf (accessed on 11 October 2019).
16. Roman, R.; Lopez, J.; Mambo, M. Mobile Edge Computing, Fog et al.: A Survey and Analysis of Security Threats and Challenges. *Future Generat. Comput. Syst.* **2016**, *78*, 680–698. [CrossRef]
17. Jacobson, V.; Smetters, D.K.; Thornton, J.D.; Plass, M.F.; Briggs, N.H.; Braynard, R.L. Networking named content. In Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, New York, NY, USA, 1–4 December 2009; pp. 1–12. [CrossRef]
18. Zhang, L.; Estrin, D.; Burke, J.; Jacobson, V.; Thornton, J.; Smetters, D.K.; Zhang, B.; Tsudik, G.; Claffy, K.; Krioukov, D.; et al. Named Data Networking. *ACM SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 66–73. [CrossRef]
19. Fan, Q.; Ansari, N. Application Aware Workload Allocation for Edge Computing-Based IoT. *IEEE Internet Things J.* **2018**, *5*, 2146–2153. [CrossRef]
20. Zhou, Z.; Feng, J.; Tan, L.; He, Y.; Gong, J. An Air-Ground Integration Approach for Mobile Edge Computing in IoT. *IEEE Commun. Mag.* **2018**, *56*, 40–47. [CrossRef]
21. Khattak, H.A.; Islam, S.U.; Din, I.U.; Guizani, M. Integrating Fog Computing with VANETs: A Consumer Perspective. *IEEE Commun. Stand. Mag.* **2019**, *3*, 19–25. [CrossRef]
22. Cao, Y.; Xu, Z.; Qin, P.; Jiang, T. Video Processing on the Edge for Multimedia IoT Systems. *arXiv* **2018**, arXiv:1805.04837v1.
23. Li, C.; Okamura, K. Cluster-based In-networking Caching for Content-Centric Networking. *IJCSNS Int. J. Comput. Sci. Netw. Secur.* **2014**, *14*, 11.
24. Saino, L.; Psaras, I.; Pavlou, G. Hash-routing schemes for information centric networking. In Proceedings of the 3rd ACM SIGCOMM Workshop on Information-Centric Networking; ICN'13, Hong Kong, China, 12 August 2013; ACM: New York, NY, USA; pp. 27–32.
25. Laoutaris, N.; Che, H.; Stavrakakis, I. The LCD interconnection of LRU caches and its analysis. *Perform. Eval.* **2006**, *63*, 609–634. [CrossRef]
26. Krishnan, P.; Raz, D.; Shavitt, Y. The cache location problem. *IEEE/ACM Trans. Netw.* **2000**, *8*, 568–582. [CrossRef]
27. Wang, J. A survey of web caching schemes for the Internet. *ACM SIGCOMM Comput. Commun. Rev.* **1999**, *29*, 36. [CrossRef]
28. Chai, W.K.; He, D.; Psaras, I.; Pavlou, G. Cache "less for more" in information-centric networks. In Proceedings of the International Conference on Research in Networking, Prague, Czech Republic, 21–25 May 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 27–40.
29. Psaras, I.; Chai, W.K.; Pavlou, G. Probabilistic in-network caching for information-centric networks. In Proceedings of the Second Edition of the ICN Workshop on Information-Centric Networking, Helsinki, Finland, 17 August 2012; ACM: New York, NY, USA, 2012; pp. 55–60.
30. Zhang, L.; Estrin, D.; Burke, J.; Jacobson, V.; Thornton, J.; Smetters, D.; Zhang, B.; Tsudik, G.; Massey, D.; Papadopoulos, C.; et al. *Named Data Networking (ndn) Project*; Relatório Técnico NDN-0001; Xerox Palo Alto Research Center-PARC: Palo Alto, CA, USA, 2010.

31. Li, J.; Wu, H.; Liu, B.; Lu, B.; Wang, Y.; Wang, X.; Zhang, Y.; Dong, L. Popularity-driven coordinated caching in named data networking. In Proceedings of the ANCS, Austin, TX, USA, 29–30 October 2012; ACM: New York, NY, USA, 2012.
32. Tyson, G.; Kaune, S.; Miles, S.; El-Khatib, Y.; Mauthe, A.; Taweel, A. A trace-driven analysis of caching in content-centric networks. In Proceedings of the ICCCN, Munich, Germany, 30 July–2 August 2012.
33. Tewari, R.; Dahlin, M.; Vin, H.M.; Kay, J.S. Design considerations for distributed caching on the internet. In Proceedings of the ICDCS, Washington, DC, USA, 5 June 1999.
34. Katsaros, K.; Xylomenos, G.; Polyzos, G.C. MultiCache: An overlay architecture for information-centric networking. *Comput. Netw.* **2011**, *55*, 936–947. [CrossRef]
35. Rosensweig, E.J.; Kurose, J. Breadcrumbs: Efficient, best-effort content location in cache networks. In Proceedings of the INFOCOM, Rio de Janeiro, Brazil, 19–25 April 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 2631–2635.
36. Lee, M.; Cho, K.; Park, K.; Kwon, T.; Choi, Y. Scan: Scalable content routing for content-aware networking. In Proceedings of the IEEE International Conference on Communications (ICC), Kyoto, Japan, 5–9 June 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1–5.
37. Zhang, K.; Leng, S.; He, Y.; Maharjan, S.; Zhang, Y. Cooperative Content Caching in 5G Networks with Mobile Edge Computing. *IEEE Wirel. Commun.* **2018**, *25*, 80–87. [CrossRef]
38. Hasan, K.; Jeong, S.-H. Efficient Caching for Delivery of Multimedia Information with Low Latency in ICN. In Proceedings of the Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), Zagreb, Croatia, 2–5 July 2019; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2019; pp. 745–747.