Caching Content on the Network Layer: A Performance Analysis of Caching Schemes in ICN-based Internet of Things

Muhammad Ali Naeem, Rehmat Ullah, Yahui Meng, Rashid Ali and Bilal Ahmed Lodhi

***Abstract*—Information Centric Networking (ICN) is a promising paradigm shift that aims to tackle the traditional Internet architectural problems and to fulfill the future Internet requirements. The traditional Internet architecture is a host-oriented architecture (i.e., TCP/IP approach) due to which the Internet of Things (IoT) have been facing issues related to data dissemination across the distant locations. Therefore, a quick comprehension to enhance the communication for improving the content transmission services is of upmost importance. To deal with the challenges of traditional IP networks, the ICN paradigm was proposed which is different from traditional IP networking in terms of (i) naming, (ii) routing and forwarding, and (iii) caching. One of the most common and important features of ICN architectures is in-network caching, which can significantly reduce content retrieval latency and improve data availability. Furthermore, in an ICN-based IoT environment, content caching at intermediate network nodes reduces the path stretch between end-users and caches the content to meet future demands. This paper compares and thoroughly investigates ICN-based caching strategies in terms of content retrieval latency, cache hit ratio, stretch, and link load, with a focus on IoT-based environments. Following a thorough simulation study, we discovered that ICN in-network caching is one of the most beneficial features for enhancing IoT-based networks.**

***Index Terms*—Internet of Things, Information Centric Networking, Named Data Networking, Caching, Latency, Future Networks.**

I. Introduction

The conventional Internet architecture is based on Internet Protocols (IP), called host oriented architecture, that focuses on identifying the communication endpoints by assigning IP address to them [1]. However, due to location dependency, data communication has been facing several critical issues such as latency and network link congestion. Moreover, applications on the Internet have continued to evolve, shifting their focus from end points to the content. Users and applications are more interested in the content itself rather than the container of the content [2]. This mismatch of traditional Internet causes a number of issues in terms of data availability, scalability, mobility, and security.

On the other hand, continuous growth of diverse applications such as file sharing, video streaming, video on demand (VoD) and user generated content (UGC) causes congestion because the network channels have limited capacity to transfer data between the distant locations [3]. Based on these insights, future Internet technologies, such as Information Centric Networking (ICN) paradigm has emerged [4], [5]. A number of ICN-based architectures have been proposed in the ICN paradigm, with Named Data Networking (NDN) gaining significant attention due to its active development and community involvement. NDN fundamentally departs from the communicational model of the conventional Internet, shifting the focus of "where" to "what" [6].

In ICN/NDN[1], all the contents are associated with unique identifier that helps to retrieve the content [7]. Moreover, the content is cached at intermediate network nodes (according to the caching policy applied) during its transmission from provider to the user to fulfill the future subsequent requests [8]. ICN/NDN comprises of many features such as name-based forwarding, consumer mobility, content security, multi-casting, and in-network caching. One of the key characteristics of ICN/NDN designs that distinguishes them from IP-based Internet paradigms is in-network caching [9], [10].

Furthermore, ICN/NDN can be thought of as one of the enabling technologies for the Internet of Things (IoT), where billions of devices will be connected and content distribution will be a big concern. [11]. However, in an IoT-based environment, content dissemination is dependent on the TCP/IP architecture, in which data is transmitted between end-to-end hosts. Consequently, the exponential increment in heterogeneous devices and the huge data transmission in IoT creates challenges due to host-centric Internet paradigm. For instance, mobility, scalability, network management, and security are most crucial for the efficient data dissemination across the Internet [12].

Although IoT is a developing technology, the current Internet design may short fall for efficient data dissemination

Muhammad Ali Naeem (malinaeem7@gmail.com) is with the School of Science, Guangdong University of Petrochemical Technology, Maoming 525000, China.

Rehmat Ullah (r.ullah@qub.ac.uk) and Bilal Ahmed Lodhi (b.lodhi@qub.ac.uk) are with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast BT95BN, United Kingdom.

Yahui Meng (mengyahui@gdupt.edu.cn) is with School of Science, Guangdong University of Petrochemical Technology, Maoming 525000, China .

Rashid Ali (rashidali@sejong.ac.kr) is with the School of Intelligent Mechatronics Engineering, Sejong University, Seoul 05006, Republic of Korea.

Equal contribution and co-first authors: Muhammad Ali Naeem and Rehmat Ullah

Corresponding authors: Yahui Meng and Rashid Ali

[1]It should be noted that we used an NDN communication model for the comparative analysis. However, we use the acronyms ICN and NDN interchangeably in this work.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

due to IP-based model. In addition, ICN/NDN paradigm provide cachable network to reduce the traffic load on content providers and brings the content near the users. The ICN/NDN in-network caching minimizes the end-to-end mapping between servers and provides location independent communication across the Internet [13].

In fact, when a mobile user's location changes in an IP-based network, the mobile user must establish new end-to-end connection. However, in ICN/NDN-based networks, the content are cached at multiple locations and can be retrieved from the nearest caching node, removing the IP-based location-centric approach. [14]. Indeed, ICN/NDN caching is an appropriate solution to provide better communication services in IoT-based scenarios which can minimize the data dissemination challenges in future Internet [15]. In literature, several papers have been published to present ICN/NDN-based IoT scenarios and the majority of the research are based on a general discussion of ICN and its benefits for IoT without detailed performance evaluation. Our research, on the other hand, provides a detailed analysis of ICN/NDN-based caching strategies for IoT scenarios. This study provides a thorough simulation analysis in terms of content retrieval latency, cache hit ratio, path stretch, and link load, all of which are well-known metrics for evaluating the performance of ICN/NDN-based IoT caching strategies.

The main contributions of our work are summarized as follows:

- We describe and discuss the state-of-the-art ICN/NDN-based caching strategies designed specifically for the heterogeneous IoT environment.
- We present a rigorous simulation analysis of several ICN/NDN-based IoT caching strategies in order to identify the best solution for the IoT environment. The investigation is carried out in terms of content retrieval latency, cache hit ratio, path stretch ratio, and network link load.
- Finally, we identify potential directions for future research and discuss solutions to these issues.

The rest of the paper is organazied as follows. Section II describes the motivation for choosing the ICN/NDN-based IoT caching schemes. Section III provides a comparative overview of the most recent related studies. Section IV is devoted to the ICN/NDN-based IoT caching strategies. In Section V, the performance evaluation of the selected caching strategies is presented using a common simulation platform. Section VI provides a summary of the analysis of selected caching strategies. Section VII presents future research opportunities and directions. Finally, Section VIII draws the conclusion.

II. Motivation for Information Centric Internet of Things Caching Approach

In the traditional Internet architectural model, all incoming requests from end users are routed to distant servers to download redundant data, significantly increasing data retrieval delay, bandwidth consumption, link usage, and server load. Furthermore, the network congestion is increasing continuously because of homogeneous data transmission due to host centric Internet approach. To this end, many solutions such as peer to peer networks (P2P) and Content Delivery Network (CDN) were proposed for efficient content dissemination. In CDN, the caching is deployed at the application layer (e.g., web caching) to enhance the content distribution. All of these proposals, however, are overlays on top of the traditional IP network. The traditional Internet employs the Internet protocol version 6 (IPv6) or IPV4 to allocate efficient address spaces to the hosts [16]. However, IPv6 has a limited amount of addresses, which can make network management and scalability difficult in the future IoT. Furthermore, IoT-based heterogeneous devices will generate large amounts of content, posing a significant problem in terms of scalability and network management.

In this regard, ICN paradigm is considered as future Internet architecture in which caching is deployed at the network layer and the application layer semantics are transferred to the network layer. A number of ICN-based Internet architectures were developed such as NDN, Data Oriented Network Architecture (DONA), Mobility-First, and Publish Subscribe Internet Technology (PURSUIT). All these architectures primarily support caching approaches such as on-path, off-path, and peer caching. In IoT-based scenarios, the caching module is highly recommended to quickly distribute the information towards the end-devices. Similarly, most of the IoT-based applications required fresh contents with respect to time constraints and most of the IoT-based content are impermanent in nature that needs to be replaced.

Over the last decade, the IoT have gained a substantial interest from both academia and industry due to its rapid expansion and flexibility to connect everything everywhere [17]. However, devices in IoT are resource-constrained in terms of storage, power, computation and communication which are essential capabilities for processing, identifying, and sensing in order to communicate in the diverse environment of IoT [18], [19]. To enhance the IoT in terms of scalability, ICN/NDN provides naming mechanisms for naming the content and the content can be accessed with the name of the content rather than location of the content [20]. Furthermore, the retrieval of content in NDN is based on a pull-based communication model, in which communication occurs only when a user sends a request for the content. Similarly, IoT devices are resource constrained, and as a result, NDN pull-based content retrieval is advantageous for the IoT-based environment in terms of reducing network resource usage.

Fig. 1 depicts an IP data dissemination scenario in which user requests are forwarded to the remote server and the same content is traversed several hops from the remote server to the user, increasing content retrieval latency [21]. Another crucial issue of IP-based Internet is the network link utilization due to the multiple transmissions of similar contents between end hosts. On the other hand, in IC/NDN, all the nodes (routers) provide cache storage to delivers the contents within short time, because in ICN/NDN, the previously requested contents are cached near the users to fulfill the subsequent requests [22], [23]. Furthermore, if the network congestion is high and the capacity of bandwidth is low, the response latency will be very high due to the amount of data traffic disseminating through a network path [24], [25].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

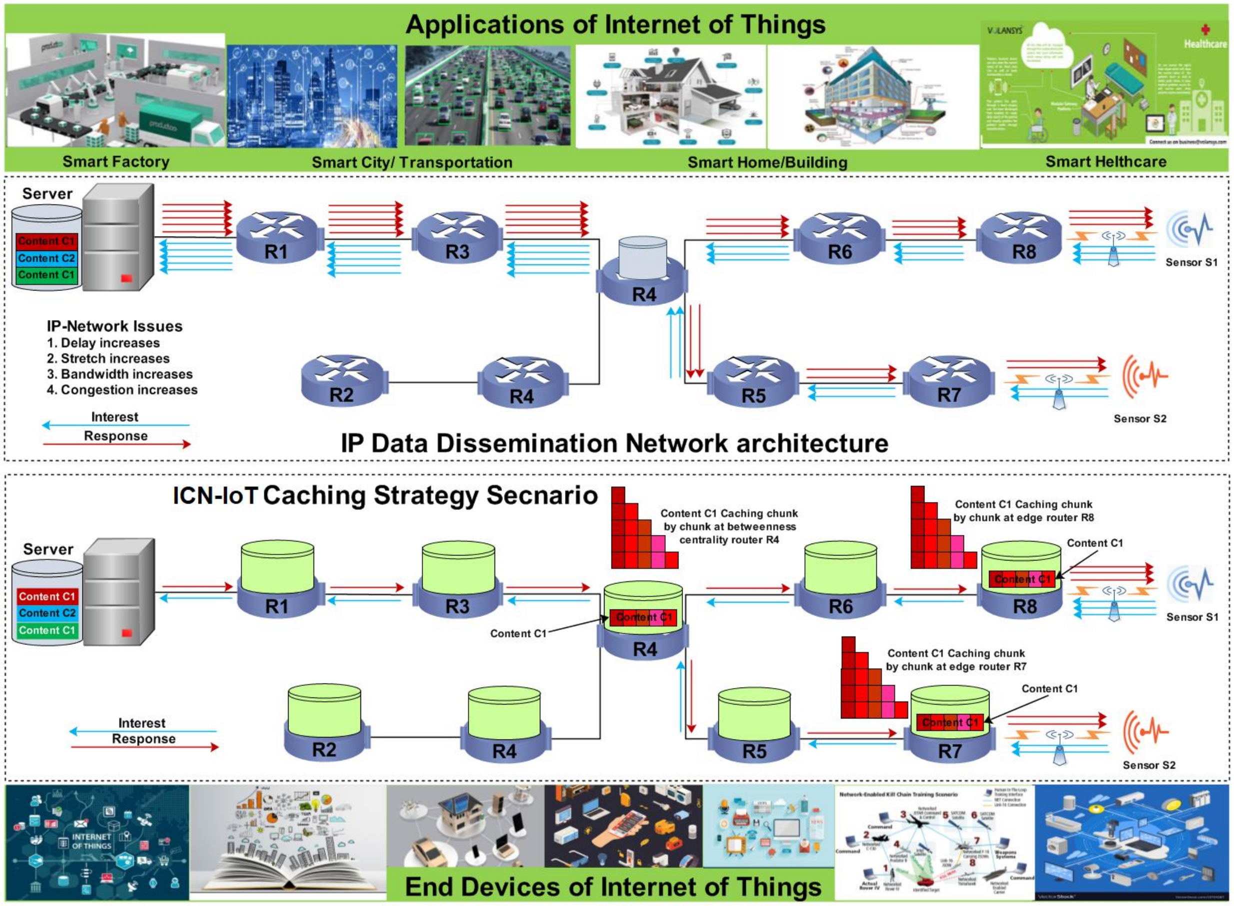


Fig. 1: IP-based verses ICN-based Data Dissemination.

In line with this assertion, the current Internet architectural design needs a paradigm shift to fulfil the future Internet requirements. Particularly, ICN Research group (ICNRG) is working on developing a communication architecture that will deliver name resolution, routing, and scalable system in ICN/NDN by deploying the cache-able network environment. ICN/NDN have better communication services over the IP-based Internet as it reduces the bandwidth consumption and link load because it provides the ability to cache content at intermediate nodes that decrease the flow congestion in network links [26]–[32]. The emergence of ICN/NDN with IoT is very promising for efficient data dissemination services and to achieve the objectives of future Internet

III. Related Work

In recent years, several surveys have been written in which the benefits of ICN/NDN are presented that can be used to improve the data dissemination in IoT-based environment. George et al. [33] discussed that how ICN's features are beneficial for IoT-based environment and how the IoT-based scenarios can achieve better performance using ICN paradigm. Lindgren et al. [34] highlighted the use of ICN in the field of IoT-based scenarios and suggests some ICN-based modules to use in IoT-based environment. However, the study presents a general overview of ICN and IoT architecture. In a study by Meddeb et al. [35], the ICN-based caching is presented as flexible approach for data transmission in IoT related environment. Moreover, a comparative study of the ICN-based caching strategies has been carried out to determine an appropriate caching policy for the IoT networks. However, this study reported the old caching strategies that were published before 2015 which is quite outdated. Recently, Arshad et al. [9] present a survey, where the ICN potentials for IoT-based environment were discussed in detail with their promising features such as routing, naming caching, mobility, and security. Furthermore, simulation tools and operating systems that were used to evaluate the performance of ICN-based IoT environment are reported in this study. However, to uphold the authors' claims of ICN for IoT, the study lacks the performance evaluation in the IoT based environment. Mars et al. [36] discussed the most popular domains of IoT-based architecture and its characteristics towards the communication technologies. However, this survey is limited to the basic introduction of ICN modules and IoT-based components. Adel et al. [37] represents a detail description about emerging scenario of ICN-based IoT and its advantage and limitations with future research directions. In addition, it differentiates the IP-based networks from ICN-based networks and suggests that ICN is a promising paradigm for IoT-based environment. However, it provides conceptual detail only and no implementation

TABLE I: Goals and limitations of existing surveys.

Year	References	Goals	Limitations
2015	George *et al.* [33]	This review explains the feature and modules of ICN-based architectures and provides background knowledge about the ICN-based IoT scenarios.	A general review is presented about ICN architectures and IoT-based scenarios without any performance evaluation.
2016	Lindgren *et al.* [34]	The objective of this study is to present an overview of ICN architecture and its components that are useful for IoT-based scenarios.	It provides a general overview of ICN and IoT architectures and their modules without any description and evaluation environment.
2017	Meddeb *et al.* [35]	This study aims to find out the most appropriate caching policy to implement in IoT-based environment.	The performances of old caching policies are reported that were published before 2015.
2018	Arshad *et al.* [9]	In this article, survey on ICN-based architectures and modules to enhance the IoT-based network environment are presented.	Lack of performance evaluation. Limited to the basic knowledge about ICN and IoT networks
2019	Mars *et al.* [36]	The most popular IoT-based domains and the benefits of ICN for the IoT-based environment were presented in this study.	It describes the basic feature of ICN such as mobility, caching, and naming and provides the limited knowledge about ICN-based IoT network environment.
2020	Adel *et al.* [37]	This study aims to provide a systematic mapping review on ICN and IoT networks. Moreover, the intersection of IoT and ICN services are also presented.	This survey is limited to the systematic review. No explanation or implementation detail were reported in this study.
2020	Naeem *et al.* [38]	This study presents the comparative analysis of an NDN popularity-based caching strategies and provides comparisons of popularity-based caching strategies via SocialCCNSim simulator.	This study is limited to an NDN-based popularity caching strategy and most of these strategies were developed before 2015. The analysis reported was evaluated in a modest platform.

detail or performance evaluation criteria are presented in this research. Recently we also evaluate the performance of ICN and specifically NDN-based caching strategies without taking into account the IoT scenarios [38]. However, in this paper, we provide our comparative analysis of the caching strategies that were especially designed for NDN-based IoT networks. Both studies use different methodologies and metrics to evaluate the performances. The previous research focused on the replications of heterogeneous content in order to identify the best caching strategy and evaluation were carried out in terms of content diversity and content redundancy. However, the focus of this research is on latency and factors that influence latency. The primary goal of this research is to examine various NDN-based IoT caching strategies with the objective of improving the overall network performance and achieving greater data availability with shortest stretch and lowest content retrieval latency. Furthermore, our focus is to find an optimal caching policy where the cache consumption, content retrieval time and the usage of resources should be reduced since these are the common issues in IP-based IoT. We also evaluate NDN-based caching strategies for IoT in terms of link load (i.e., link utilization). The comparison of existing works with our work is shown in Table I.

IV. NDN-BASED IOT CACHING STRATEGIES

To provide an efficient and reliable data dissemination using self-certifying content security, NDN delivers named-based contents routing and caching facilities to improve the data accessibility across the Internet [39], [40]. Due to an efficient and fast data delivery, NDN is considered as prominent network model to improve the overall performance of IoT-based environment [41]. Although IoT is a promising approach for the efficient data dissemination, it has been facing several critical issues such as content retrieval latency, data availability, and stretch (path between the publisher and subscriber) [42]. NDN, on the other hand, provides in-network caching to improve the data transmission by caching transmitted content at intermediate locations along the subscriber path. As a result, the content retrieval latency is reduced and the data availability is maximized [43], [44]. The subsequent subsections present NDN-based caching strategies and are discussed as follows.

A. Fixed and Mobile Converged FMC

Fixed and Mobile Converged (FMC) [45] was developed to provide location-independent named-based and fast data dissemination in fixed and mobile Internet as well. The most significant module of this scenario is functional node which works as a Unified Access Gateway (UAG) and is used to provide services to the fixed as well as mobile network in an IoT-based environment as shown in Fig. 2. The UAG is responsible to cache the content within the FMC network and transmit the contents to the end-users. Moreover, a copy of transmitted content is cached near the users to fulfill the subsequent users' requests. A cache controller is used in conjunction with UAG to provide services to the requests and contents of users. For instance, when a user sends a request to the network, the network forwards the incoming request to the UAG, and then UAG sends this request to the appropriate publisher. The FMC uses several replacement policies to evict the content as the cache of the network node overflows.

Fig. 2 depicts the FMC scenario in which two types of networks emerge (Fixed and Mobile). In the given scenario, user U1 sends a request to download the content C3. When the request from user U1 arrives at node N5, UAG forwards it to the appropriate publisher (publisher P1). As a result, the publisher P responds to the request by sending the corresponding content to the user U1 and a copy of transmitted content is cached at node N5 to fulfill the future subsequent users' requests.

The aim of FMC is to maximize the caching efficiency at the edges (subscriber's nodes). However, it increases the communication overhead because in FMC, the UAG sends maximum number of data packets to the caching nodes that increase the link overhead. On the other hand, the shared caching system in FMC increases the caching efficiency at user sides by using multiple replacement policies.

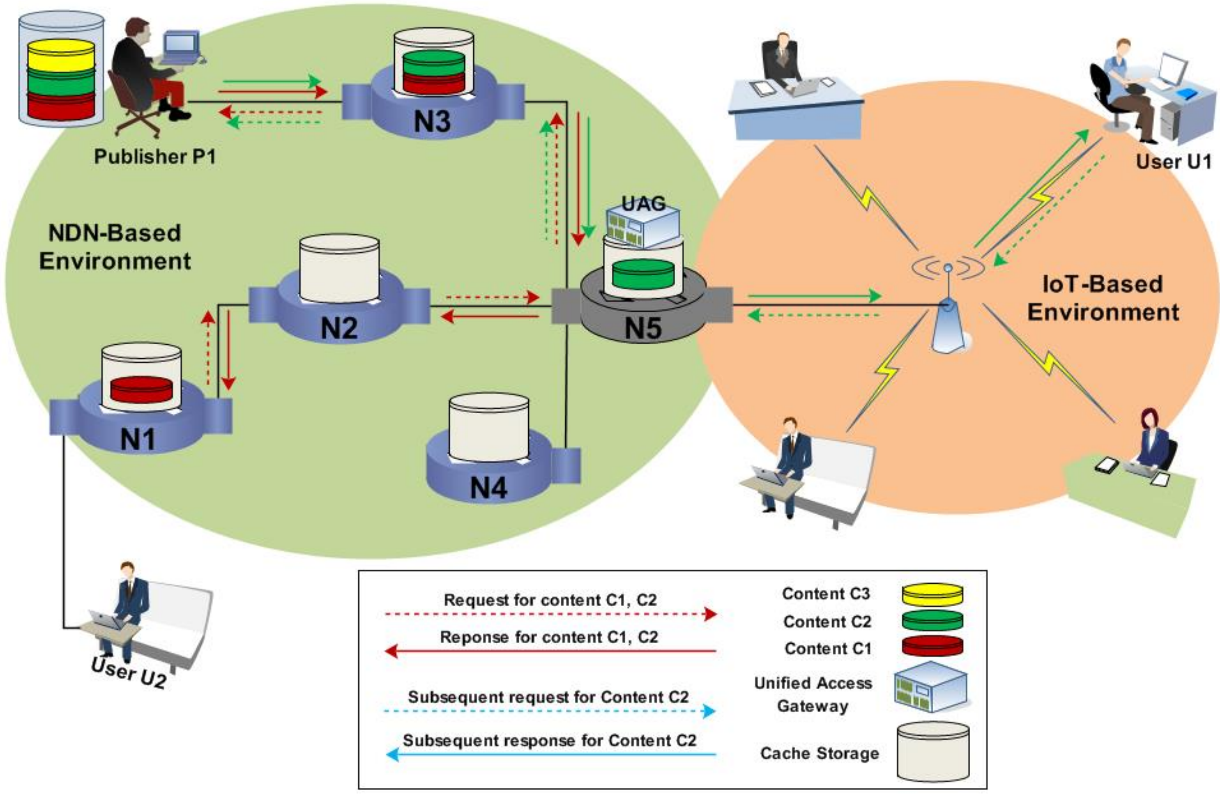


Fig. 2: Fixed and Mobile Converged Scenario.

B. Client-Cache Strategy (CCS)

Client Caching Strategy (CCS) [46] is a coherence-based caching strategy in which the validities of cached content is checked to find the popular content for further caching operations. The content selection and caching mechanisms in CCS depends on three phases such as content caching, caching method, and caching node reduction. In primary phase, the on-path caching approach is used to cache the transmitted contents within network nodes along the subscriber's path. Unlike off-path caching approach, it does not include supplementary metadata to inform the users to decide which network node is suitable to cache the disseminated content to fulfill the subsequent requests of the desired users. In second phase, an appropriate network node is selected that can reduce the distance from the requested users (the user has sent their requests for similar contents) and cached contents. Since the CCS selects the node which is associated to the highest number of neighbor nodes (known as betweenness centrality node), the betweenness centrality node has the ability to significantly reduce the path length between the users and the cached content. In third phase, the requested contents are cached at selected centrality node which is located close to the network edge nodes.

The goal of CCS is to maximize the content's validity that is the reason CCS is known as coherent caching strategy. The validity of a content is measured as the requested content founds within the network cache. Consequently, the content is suggested as valid, if the validity of cached content is smaller than the validity of same content at provider node (publisher). Fig. 3 illustrates the content selection and content caching mechanism in CCS where multiple requests are sent from the sensor S1 and sensor S2 to download the content C1 and C3 from publisher P. As shown in Fig.3, validity of content C1 at publisher P is equal to 6 and the validity of content C3 is equal to 5. However, the validity of content C1 at caching node (betweenness centrality node) is 4 and the validity of content C3 at caching node is 3. Therefore, according to the CCS, the validities of requested content C1 and C3 at publisher is higher as compared to the validities of content C1 and C3 at caching node. Thus, the content C1 and C3 are suggested to be cached as caching node (N4) as shown in Fig. 3. On the other hand, the content C2 has lower validity at publisher P as compared to validity of C2 at caching node N4. Therefore, the content C2 is not recommended to be cached at intermediate node (N4). The N4 is selected as caching node because it has maximum connections with neighbor as compared to any other node along the data routing path. Hence, all the subsequent requests are accomplished at caching node N4.

CCS aims to improve the validities of contents and achieves lower performance in terms of latency. In CCS, the contents' validities are observed and measured whenever a request is received. The betweenness centrality node delivers high performance, while it results in congestion as the cache of centrality node becomes full. However, if a content is most popular and its validity at caching node becomes higher as compared to its original validity at the publisher, the content will be deleted from the intermediate node. Therefore, a large number of subsequent requests need to be forwarded via

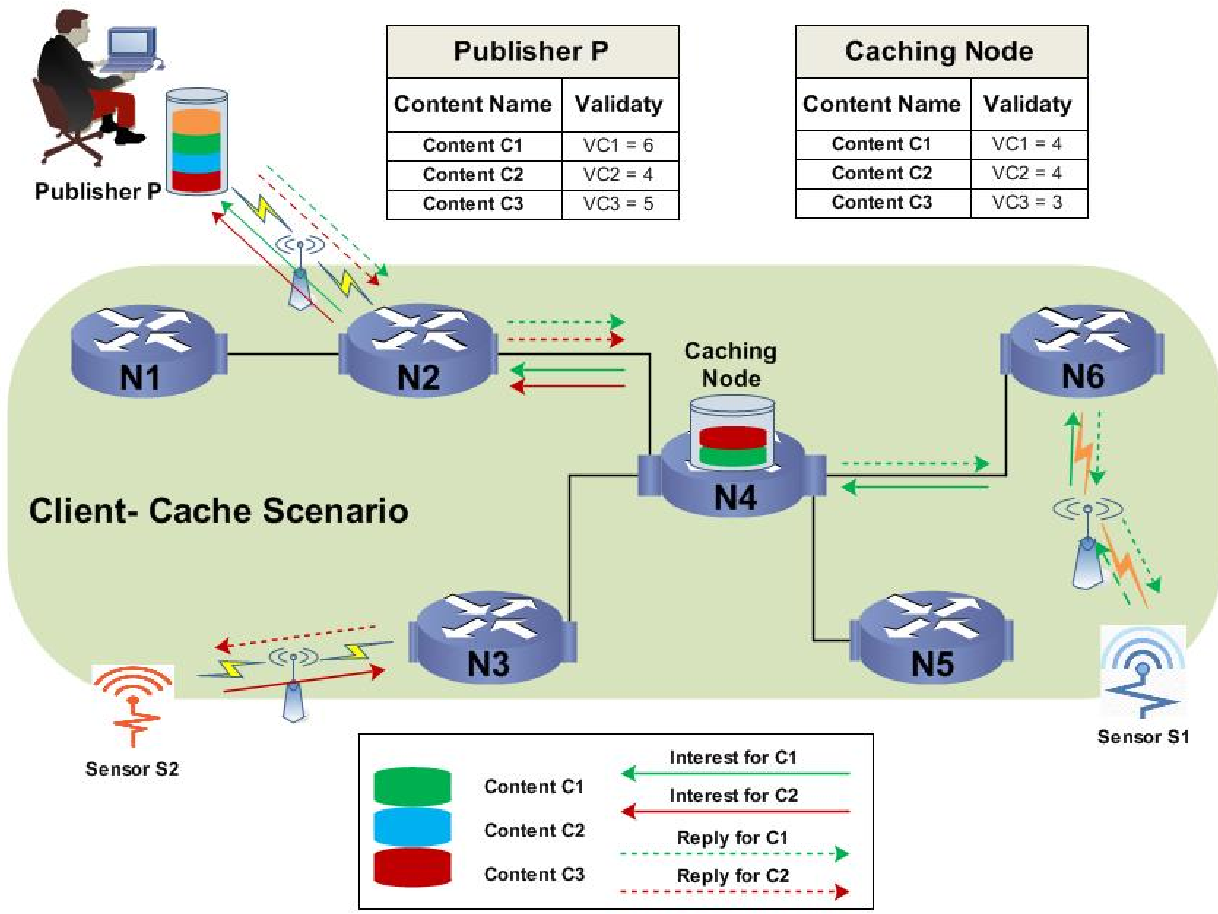


Fig. 3: Client Cache Strategy Scenario.

several hops to the publisher node. As a result, the overall caching performance is reduced.

C. Sleep-based Caching Strategy (SCS)

The Sleep-based Caching Strategy (SCS) [47] was developed to minimize the usage of energy in an NDN-based IoT environment by caching the most recently requested content at the intermediate network nodes to fulfill the subsequent requests. SCS tries to improve the sleeping mode of a node, content caching mechanism, and content replacement mechanism. The sleeping mode was further divided into coordinated sleeping mode and uncoordinated sleeping mode.

The coordinated sleeping mode is responsible to make relationship among the nodes to keep the IoT nodes inactive for particular time to place and replace the transmitted content. Consequently, an IoT node can cache content, when it is in the state of inactive mode and in active mode known as deputy node. When a node becomes a deputy mode, it sends request message to its neighbor nodes to send content for caching in its content store. Multiple nodes can become deputy nodes when the size of transmitted content is larger than the free cache space.

On the other hand, the uncoordinated sleeping mode is responsible for changing the mode of an IoT node. It decides a particular time t for the active and sleep mode of a node. The node caches a content with probability of 0.5 for specific time being in active mode. For the content replacement mechanism, a name-based content replacement policy is used termed as Max Diversity Most Recent (MDMR). In SCS, another replacement policy is developed to replace the content from a caching node known as Prioritized- Max Diversity Most Recent (P-MDMR). Usually, in NDN the content names are presented in prefix and suffix order in which the prefix defines the sensor while the suffix defines the time stamp for content during caching mechanism. Therefore, the P-MDMR content replacement policy works on the name for a particular cache such as a heat node used to store the temperature in its content store. Moreover, to release the cache, the content with non-prioritized prefix will be replaced.

In Fig. 4, user U1 and user U2 send two requests to retrieve content C1 and content C2 from publisher P. The publisher P sends the corresponding contents to the user U1 and user U2. During the transmissions of content C1 and content C2, the Node N4 became deputy and sends the request messages to cached content C1 and C2. Therefore, the subsequent requests from user U3 and user U4 are accomplished from deputy node N4 since the content C1 and C1 already cached at node N4.

The SCS aims to enhance two mechanisms such as node sleeping mechanism to improve the utilization of limited energy resources and the management of limited cache storage of the IoT devices. SCS caches the popular contents at

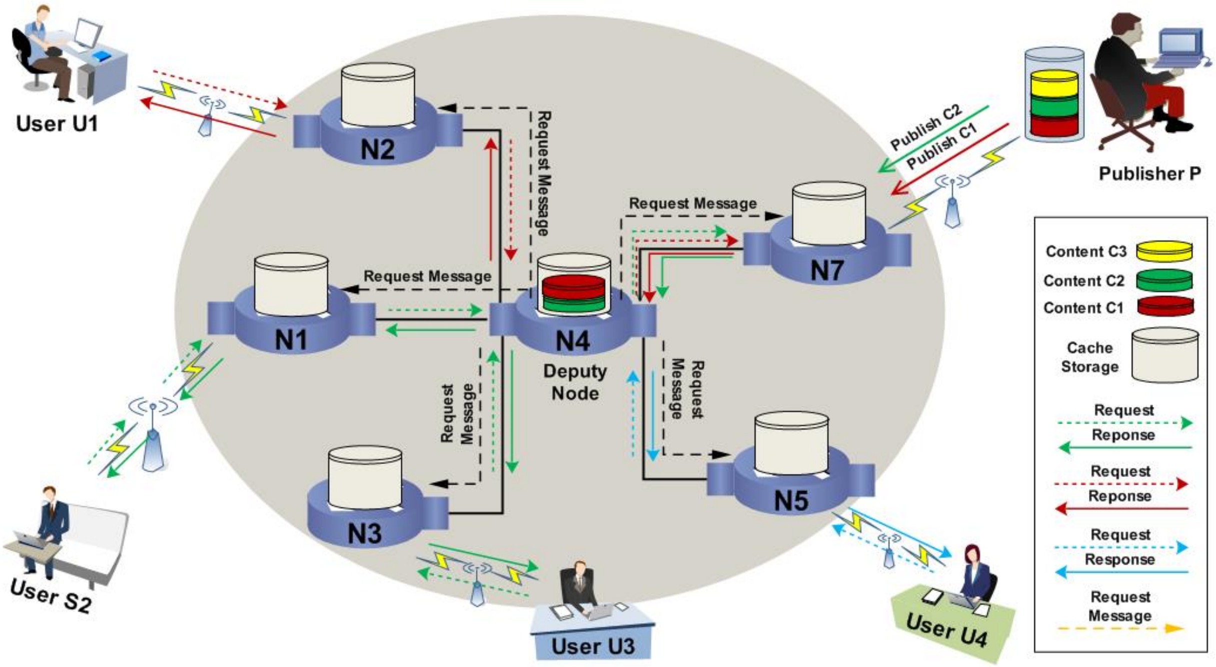


Fig. 4: Sleep-based Caching Strategy Scenario.

multiple deputy nodes and when caching node is in sleeping mode, the content is accessed from the other deputy node(s). Consequently, a smaller number of requests may be sent to the main publisher and thus, the overall caching performance gets reduced and cache hit is improved.

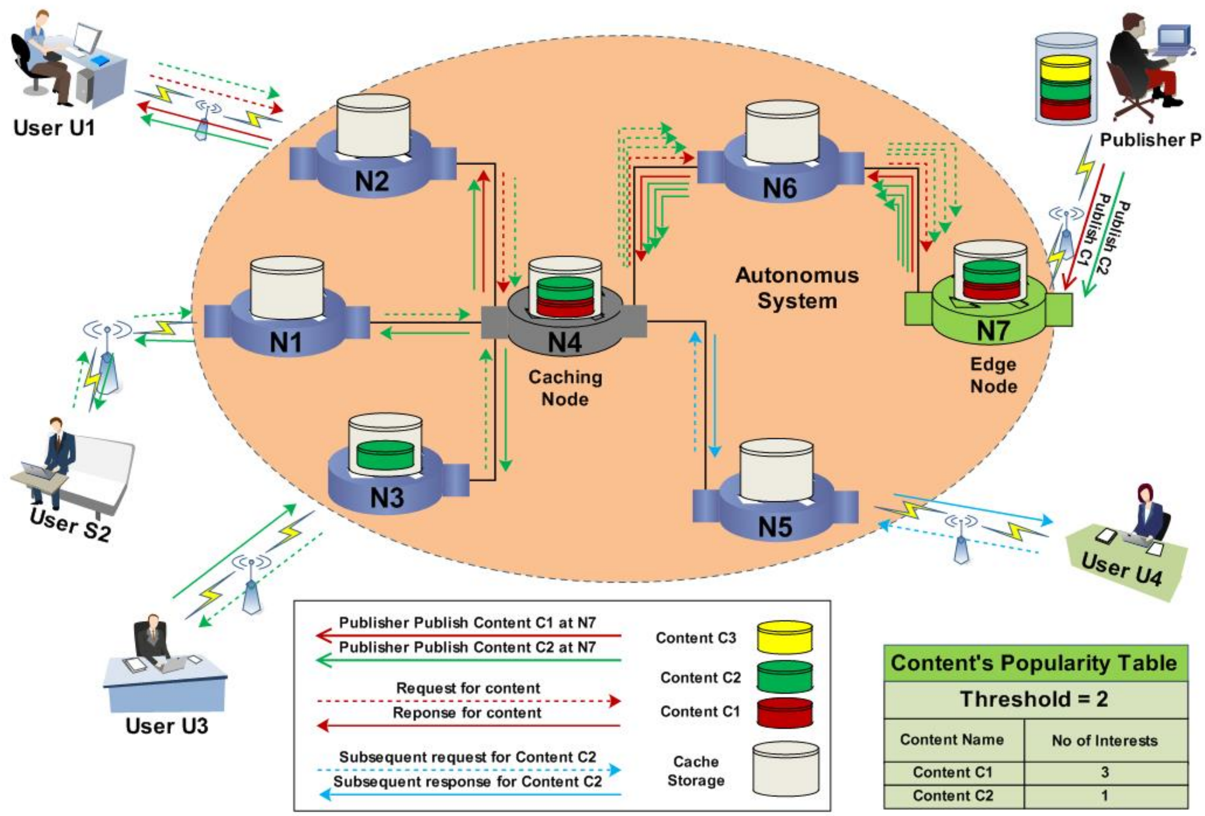
D. Periodic caching strategy

Periodic Caching Strategy (PCS) [48] was proposed to enhance the content dissemination through maximizing the availability of desired information for the end users. Basically, it was developed to increase the cache hit ratio with minimum content retrieval latency. Moreover, PCS tries to reduce the length of publisher-subscriber path. In PCS, each network node is associated to a statistics table to store the content name, request frequency counter, and a threshold value. Each node locally counts the number of incoming users' requests and separate them into a table with respect to the content name. Consequently, if a content name receives the number of incoming requests more than or equal to the threshold value, the content is labeled as popular one. Therefore, the popular content is recommended to be cached at intermediate network nodes between the requested user and the publisher.

All popular contents initially cached at the edge nodes of an Autonomous System (AS), but when the caches of these nodes become full, the Random Replacement Policy (RRP) is followed for content eviction. Random policy is used because its computational complexity is O(1) [49]. Therefore, the searching overhead during content eviction process will be quite minimum. After that, the evicted contents follow betweenness centrality in each AS, i.e., they are cached at a node (in each AS) that maximum nodes traverse it during content downloading. In this way, all the requests pass through the node having maximum betweenness centrality and if there is already cached content, the incoming requests are satisfied from the cache rather than forwarding to the server.

In addition, if the requested contents are not found at betweenness centrality node, the requests are forwarded to the edge node and the edge node replies with the actual content. By doing so, the several goals are achieved such as (i) the cache hit rate and content diversity are improved, (ii) the eviction rate is kept at minimum, and in turn the content retrieval latency is reduced, (iii) and the overall bandwidth is utilized in an efficient manner. The content selection and content caching scenario of PCS is illustrated in Fig. 5 where multiple requests are received from user U1, user U2, and user U3 to download content C1 from edge node in an AS. Initially, the publisher P had published the content C1 in network and it was cached at the edge node N7 in an AS. Therefore, the content C1 has received a greater number of requests than threshold. Consequently, it is labeled as popular one and the content C1 has to be cached at intermediate nodes with respect to PCS caching criterion.

PCS implements the ASs to divide the whole network into a number of small domains to deliver most desired contents near the end-users that fulfill the subsequent requests. Besides, PCS implements the centrality-based caching without affecting the contents' popularity. However, PCS caches the content at the edge node and centrality node in an AS. Therefore, path becomes congested and most of requests cannot be fulfilled at centrality node and needs be to forwarded to the edge node to retrieve the desired content. As a result, the network performance is disturbed.



Content's Popularity Table	
Threshold = 2	
Content Name	No of Interests
Content C1	3
Content C2	1

Fig. 5: Peridioc Caching Strategy Scenario.

E. Tag-based Caching Strategy (TCS)

The Tag-based Caching Strategy (TCS) [50] usually use tag filters for the lookup method in order to exactly match the forwarding contents to the appropriate location. In TCS, all the network nodes have specific type of tag list that is used to identify the frequently requested contents to be cached at intermediate network nodes. Tag list provides tag filters that are used to generate hash function to improve the content dissemination. When a user request arrives at a network node, it decides using corresponding tag filter whether the content is cached. All the tags are mapped and hashed in a counter to find the highly requested content. If the counter of tags become equal or higher than the threshold value, the content is recommended as most requested (popular) and to be cached at intermediate nodes. Therefore, all the network nodes along the data delivery path check the tags to find the suitable location for content caching.

The content selection and content caching mechanism of TCS is illustrated in Fig. 6, where multiple requests are sent from sensor S1 and sensor S2 to retrieve the content C1 and C3 from the publisher P. According to TCS, the threshold value is suggested as 2 (given in publisher P table). Therefore, the content C1 and C3 are recommended as popular with respect to the given threshold value. As a result, both content C1 and C3 is suggested to be cached close to the requesters to fulfill their subsequent requests. Hence, content C1 and C3 are cached at node N7 and node N6 which are the closest nodes to the requested sensors S1 and S2. On the other hand, the content C2 has received only one request as shown in publisher table and thus it is not recommended to be cached at intermediate nodes.

TCS caches the desired content at the network edge nodes that significantly minimizes the latency. However, it cannot guarantee that the edge node will significantly improve caching performance; rather, it is dependent on the scenario, as the most popular content may be cached at the edge node while subsequent requests must traverse several hops to retrieve the content from remote locations. The PCS improves the performance by caching the popular contents at the most visited location (betweenness centrality node). Therefore, a large number of subsequent requests are accomplished from centrality nodes.

F. Packet Update Caching

The Packet Update Caching (PUC) [51] strategy was developed to improve content caching in an ICN-based IoT network. In PUC, content caching is based on the clustering of a set of nodes with a variety of features and resources that work together to execute tasks such as parallel processing and load balancing. The cluster is made up of using cluster head (CH) and cluster member (CM). Furthermore, in each cluster, the CH is determined to perform content caching operations. The CH acts as a link between the Base Station (BS) and the interconnected nodes, allowing users' requests and contents to be transmitted. The CM, on the other hand, is used to transmit user requests and content from CH to end-users.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

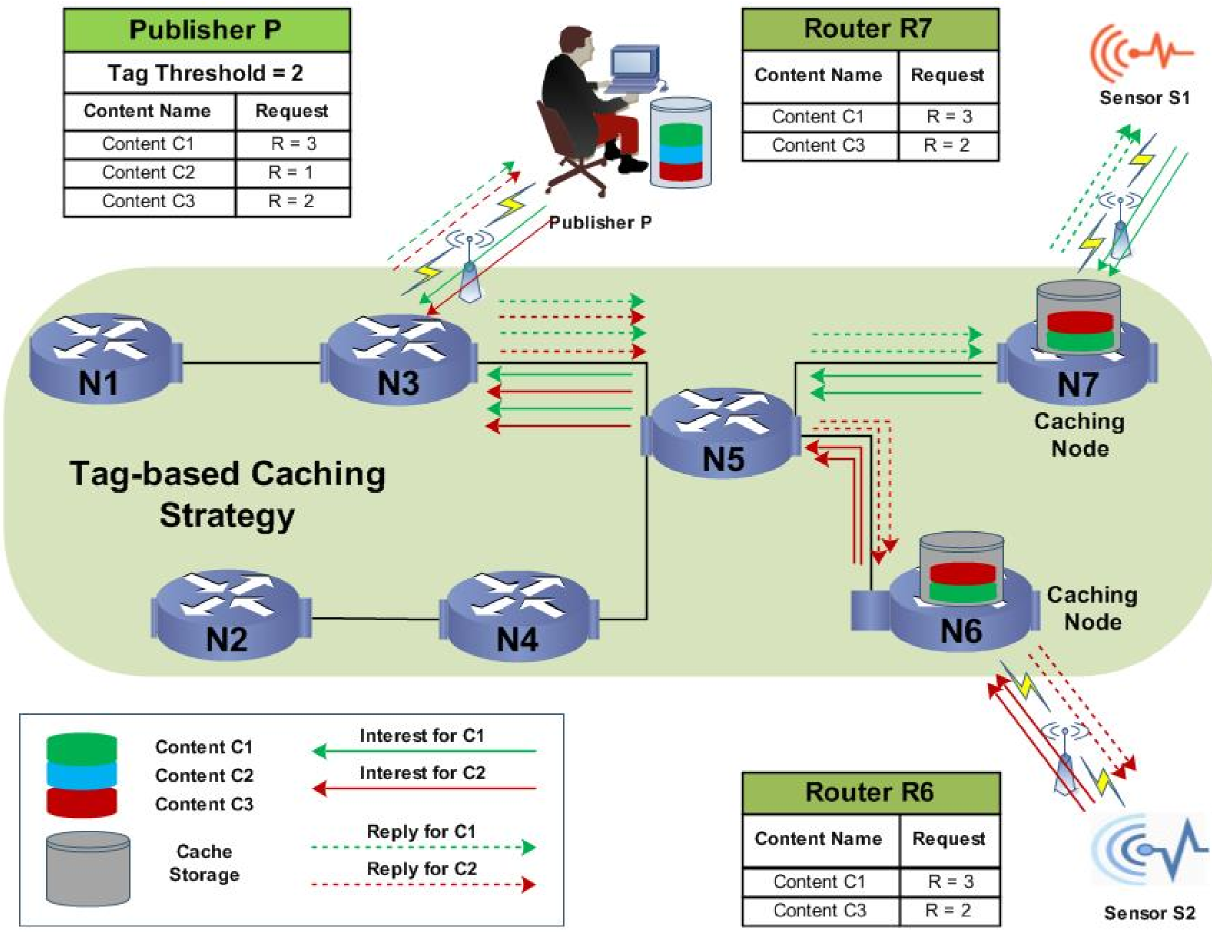


Fig. 6: Tag-based Caching Strategy Scenario.

In PUC, CH is chosen based on three factors: the number of neighbours, residual energy, and the distance between the requesting nodes and the BS. The workload of each node is determined by measuring the node's degree, capacity, and power level. Consequently, the workload is used to find the CH, and the CH is used as circular buffers to store the users' requests and contents. In order to accommodate newly arriving contents, the data purging technique is used in PUC for the permanent deletion of old contents from the cache memory. Furthermore, a cloud-based incapsula technique is used for content freshness, and it protects the data from attacks. Incapsula provides a content switching refresh scheme as well as updated data for ICN-based IoT caching.

Fig. 7 illustrates the content caching mechanism of PUC in which a User U1 sends a request to download Content C2 from Publisher P. The request is routed to the publisher via CM 1, CH3, CH2, and CH2. The Publisher P instantly responds to the request and sends the Content C2 to the User U1 and a copy of C2 is cached at CH3. As a result, the subsequent requests can be fulfilled from CH3. The requested content is cached locally at the CH of that zone from where the request was generated. When the content is updated or replaced, the circular buffer and incapsula procedure are invoked.

V. PERFORMANCE EVALUATION

A. Simulation Environment

In this section, we discuss the simulation environment for analysing FMC, SCS, PCS, CCS, TCS and PUC. To evaluate the performance of aforementioned caching strategies, the Icarus [52]–[57] discrete event-based simulator is selected, that is a specific caching simulator for ICN/NDN. Moreover, we select GEANT [58], [59] and WIDE [60], [61] topologies for our simulation. The GEANT is known as real world topology having 40 nodes with 61 edges and WIDE is known as Japanese network that comprises of 30 nodes with 33 edges. GEANT is an academic network spread around the globe and WIDE network was first established network in Japan. To evaluate the performance, the caches are warmed up with 50,000 requests. Basically, the subsequent users' requests are used to evaluate the caching performance. For determining the performance of in-network caching in NDN, we used the Zipf-distributed content popularity model with skewness parameters of α as 0.6 and 1.0, indicating low and high popularity models, respectively. [62], [63]. The content universe (number of contents) is set as F=10,000 and the network cache size is varied from 2% to 20% of total number of contents. The main simulation parameters are summarized in Table II.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

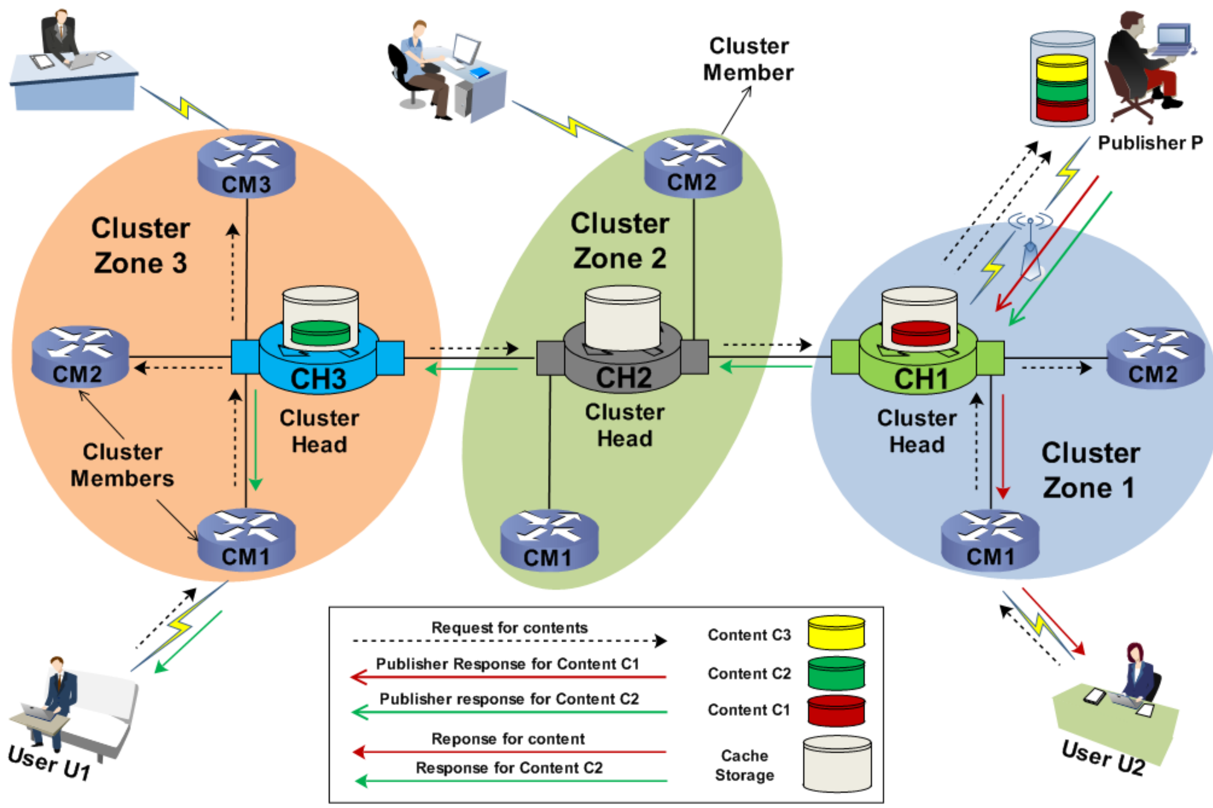


Fig. 7: Packet Update Caching.

TABLE II: Simulation Parameters.

Parameters	Values
Number of warm-up requests	50,000
Number of measured requests	50,000
Content Popularity Model	0.6 and 1.0 (low popularity model and high popularity model, respectively)
Number of contents	10,000
Request rate	1 request/second
Cache-Size	0.02~0.20
Topologies	GEANT, WIDE
Performance metrics	Latency, Cache-Hit Ratio, Stretch, and Link Load

B. Performance Metrics

For performance evaluation, we select four basic performance metrics such as content retrieval latency, cache hit ratio, stretch, and link load. These metrics are broadly used to evaluate the performance of NDN-based IoT networks.

1) Content Retrieval Latency: Content retrieval latency refers to the time required to complete one dissemination process of a request from a user to the provider node and from the provider node to the user (i.e., transfer of requested content from provider to the interested user) [64]. Usually, in NDN-based IoT caching, the latency is measured in milliseconds and can be defined as follows:

$$Latency = request\,travel\,time + Content\,travel\,time \quad (1)$$

2) Cache Hit Ratio: Cache hit ratio is a measure of how many interest packets a cache can satisfy, compared to how many interests are forwarded in the network [65]. The higher cache hit rate indicates the greater number of requests is accomplished from network and can be defined as follows:

$$Cache\,Hit\,Ratio = \frac{Cache_{hits}}{Cache_{hits} + Server_{hits}} \quad (2)$$

3) Stretch: Stretch refers to the distance covered (i.e., number of hops traversed) by a user request to find an appropriate content from in-network nodes [66] and can be defined as follows:

$$Stretch = \frac{\sum_{i=1}^{R} Hop - travelled}{\sum_{i=1}^{R} Total - Hop} \quad (3)$$

where $\sum_{i=1}^{R} Hop - travelled$ represents the distance in hops traversed by a request between the requested user and content caching node, r shows a user request, $\sum_{i=1}^{R} Total - Hop$ denotes the total distance covered by multiple requests, and R shows the total number of received requests. If the distance between user and the caching node is small, the requested content is cached near the user. Therefore, it provides a small stretch path to download the desired content [67].

4) Network Link Load: Link load is a well-known performance metric that is used to measure the bandwidth utilization on a network link. Network link load shows the number of bytes used in a network link in order to transfer a user request from user to the publisher or the number of bytes used for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

content transfer from publisher to the users in unit time and can be defined as follows:

$$Linkload = \frac{Req_{size} \times Req_{Links} \times Con_{size} \times Con_{Links}}{Time} \quad (4)$$

where time shows the durations which is required to send a request to the publisher node and the corresponding content to the user, Req_{size} and Con_{size} represents the size of user request in bytes and the size of requested contents in bytes respectively. The Req_{Links} and Con_{Links} denotes the number of network links that are used to transmit the request to the publisher node and the transmit the content to the user. Therefore, the link load can be defined as, the number of bytes transmitted on a network link to retrieve the content from publisher.

C. Results and Disussion

In this section, we present the main simulation results as well as a discussion of the performance analysis.

1) Discussion on Content Retrieval Latency: Fig. 8 (a, b, c, d) illustrates the latency performance on different topologies (e.g., GEANT and WIDE) using different skewness parameter (e.g., α = 0.6, and 1.0) and various cache sizes (e.g., 0.02~020). It is evident from Fig. 8 (a, b, c, d) that the PCS outperformed in terms of content retrieval latency. The reason is that, the PCS implements the ASs to divide the whole network into a number of small domains to deliver most desired contents near the end-users that fulfill the subsequent requests. As we increase the cache size and content popularity value, the performance gets better.

However, the CCS performs worse than all other strategies, because the validities of contents need to be checked each time when a request is received. Consequently, due to huge number of requests generation, the validities of contents are checked which in turn increases the response latency. Hence, the CCS takes more time to respond to the user's requests. Moreover, the FMC performs somehow better than CCS with both topologies because it provides shared caching system to the users and the content are cached closer to the users that reduces the latency for subsequent requests. In addition, the SCS tries to improve the performance by caching the most recently downloaded contents at intermediate nodes that increases the data availability in short time period. As a result, the latency is reduced.

On the other hand, the TCS minimizes the content retrieval latency for subsequent requests due to caching content at the network edges. The PCS improves the performance by caching the popular contents at the most visited location (betweenness centrality node) and PUC caches the content at nearest CH. Therefore, a large number of subsequent requests are accomplished from centrality nodes and CH. As a result, the performance improves.

2) Discussion on Cache Hit Ratio: Fig. 9 (a, b, c, d) illustrates the caching hit performance of selected strategies with respect to different topologies and popularity parameters such as 0.6 and 1.0. It is can be observed, that the performance of all caching strategies is increasing as the cache size expands. The reason is that, when we increase the cache size, the more content can be accommodated in the nodes' caches and therefore, a large number of incoming requests are accomplished from the intermediate network nodes.

At the small cache size (0.02) and low popularity (0.6), the PCS and TCS have similar performance of cache hit rate. However, the PCS performs much better as compared to the CCS with respect to the large cache size. Similarly, the SCS also performs better than CCS with large cache size. Moreover, the FMC and CCS perform linearly with respect to cache size, but the CCS achieves worse cache hit ratio as compared to other caching strategies. We have observed that the PCS achieves better performance than all other caching strategies because it increases the amount of most accessed contents by caching them at centrality node in an AS, where maximum requests are traversed and accomplished. Indeed, the PCS provides additional facility of moving content from edge node to the betweenness centrality node without affecting the contents's popularity. Thus, if a content has a good popularity value in recent and when it moves from an edge node to the betweenness centrality node, the popularity will remain same. Hence, a large number of incoming requests will fulfill from the betweenness centrality node.

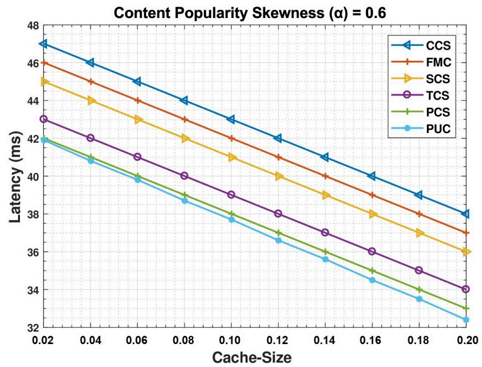
Similar to PCS, TCS also achieves better cache hit performance. The reason is that, it caches the requested contents at the network edges for fulfilling the subsequent requests. However, the CCS achieves low cache hit ratio because it considers the validity of a content to perform caching operations that takes extra time to calculate popularity of a content and hence, the cache hit rate is decreased.

Furthermore, the PUC has higher cache hit performance because it shortens the distance by caching the content at CH to meet the demands of subsequent users' requests.

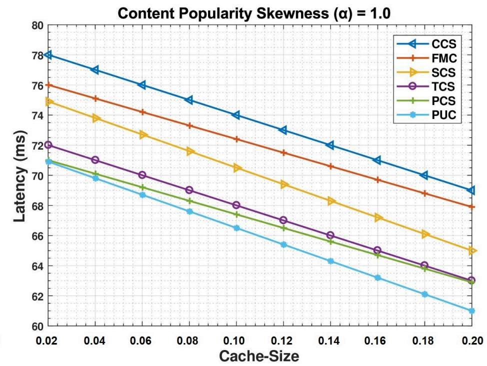
3) Discussion on Stretch: Fig. 10 (a, b, c, d) depicts that PCS perform better in terms of small stretch path. The reason is that, PCS has the ability to cache transmitted content near to the users due to its betweenness centrality within an AS that reduces the path stretch from user to the provider node. Moreover, PCS always have backup centrality node to provide cache for the most recently accessed contents and hence, the path stretch for the subsequent transmission of requests and content get reduced. Therefore, PCS is effective than other strategies in terms of stretch to provide short distance in content retrieval process. On the other hand, CCS has produced a smaller amount of stretch with respect to all cache sizes and popularity parameter (e.g., α = 0.6 and 1.0). In this strategy, a number of requests need to traverse to the main source when the cache overflows at the betweenness centrality node alongside the data downloading path. However, FMC and SCS relatively perform better than CCS with all selected parameters. Nevertheless, both have performed less comparatively to PCS and TCS.

On the other hand, the CCS measures the contents validity at multiple locations (at publisher node and intermediate node(s)) to perform caching operation that increases the path stretch. The reason is that, if a content is very popular and its validity at caching node becomes higher as compared to its original validity at the publisher node, the content will be

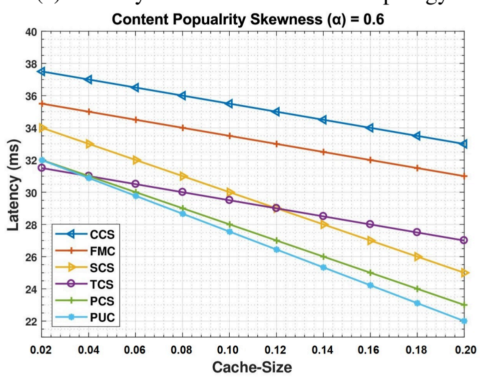
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal



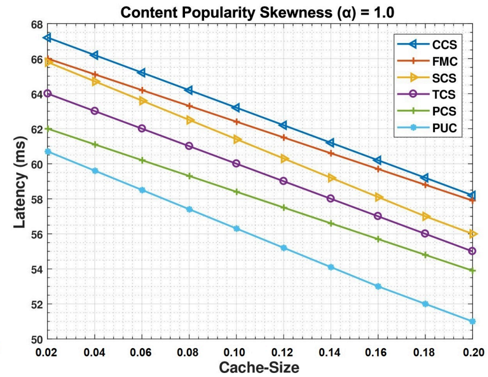
(a) Latency measure on GEANT topology



(b) Latency measure on GEANT topology

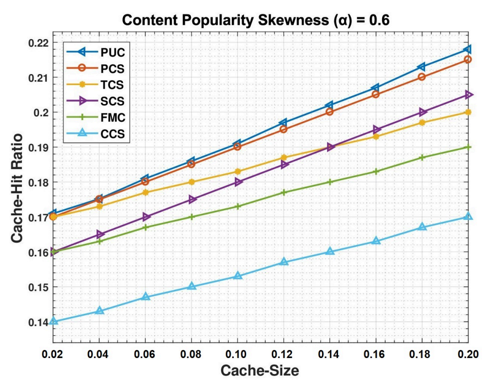


(c) Latency measure on WIDE topology

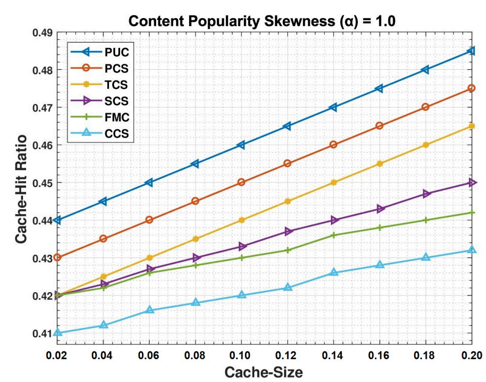


(d) Latency measure on WIDE topology

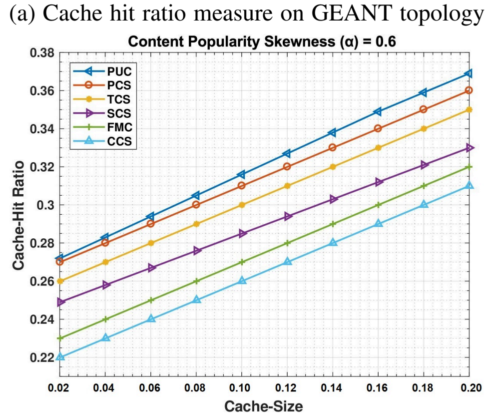
Fig. 8: Latency measure on different topologies for different popularity models.



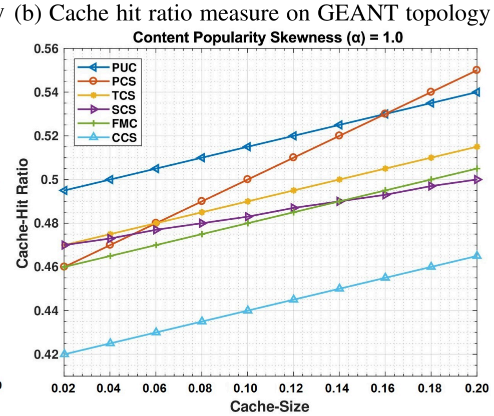
(a) Cache hit ratio measure on GEANT topology



(b) Cache hit ratio measure on GEANT topology



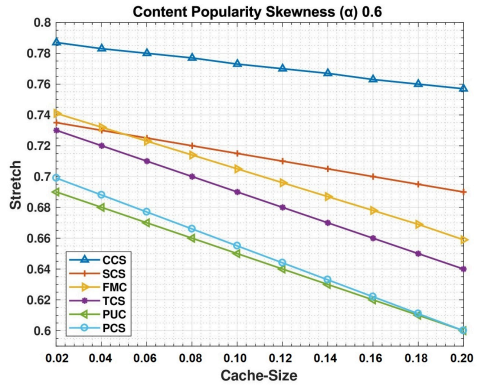
(c) Cache hit ratio measure on WIDE topology



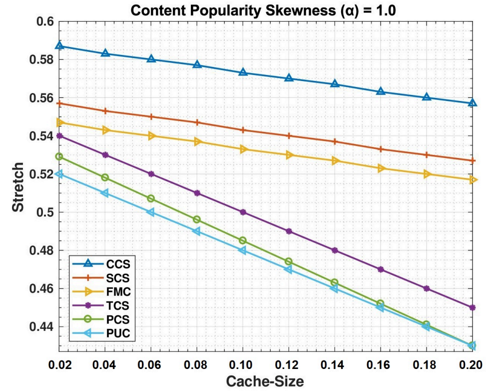
(d) Cache hit ratio measure on WIDE topology

Fig. 9: Cache hit ratio measure on different topologies for different popularity models.

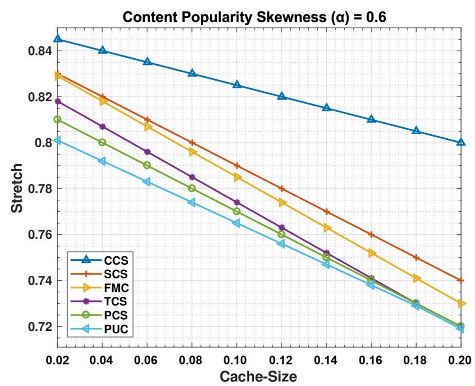
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal



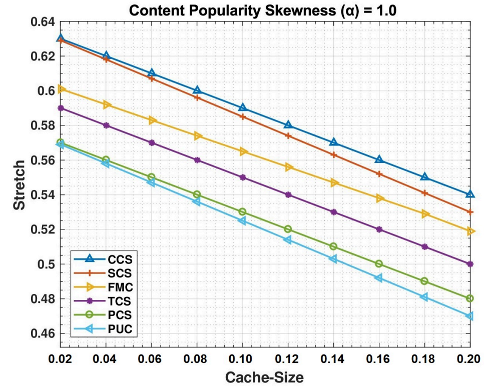
(a) Stretch measure on GEANT topology



(b) Stretch measure on GEANT topology



(c) Stretch measure on WIDE topology



(d) Stretch measure on WIDE topology

Fig. 10: Stretch measure on different topologies for different popularity models.

deleted from the intermediate node. Thus, a large number of subsequent requests need to be forwarded via several hops to the publisher node. Hence, the overall stretch ratio is decreased. In addition, the TCS performs better to reduce the path stretch for the subsequent request by caching the content at edge nodes. However, if a popular content is cached at edge node, the subsequent requests from other nodes cannot access that content. Consequently, the incoming requests from other edge nodes need to traverse several hops to retrieve the same content from the publisher node. Therefore, the overall stretch performance gets reduced.

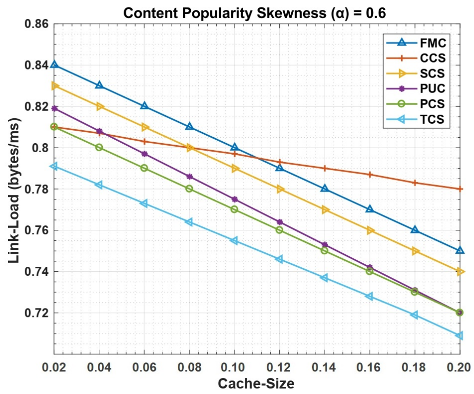
However, PUC reduces the stretch between end users and the cached copy of content at CH. Therefore, it shows better performance in terms of stretch.

4) Discussion on Network Link Load: We have performed simulation for link load using two popularity models such as low popularity model with popularity skewness α =0.6 and high popularity model with popularity skewness α=1.0. Fig. 11 (a, b, c, d) shows that the link load performance increases with the cache size because all the caching nodes have extra storage with large cache size to accommodate the requested contents near the user. FMC perform poor than other strategies because, it sends maximum data packets using UAG node on a network link that maximizes the load on a link. In FMC, the UAG node is used as gateway to deliver data dissemination services to the mobile and fixed users in an IoT-based environment. On the other hand, the SCS perform better to some extant as compared to FMC because, SCS caches the popular content at multiple deputy nodes and when a caching node is in sleeping mode, the content is accessed from another deputy node.

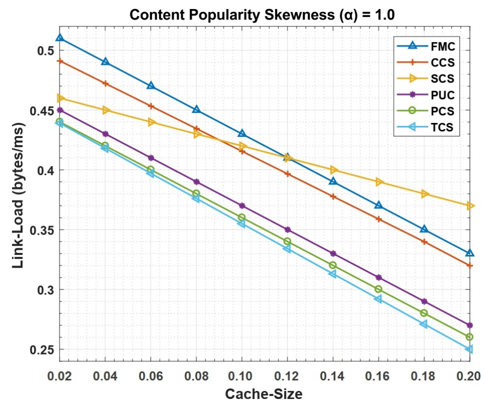
In CCS, the contents are cached at betweenness centrality node and network edge nodes. Sometime, the betweenness centrality node delivers high performance, sometimes it got congested as the cache of centrality node becomes full. Therefore, it shows moderate performance in terms of link load. In addition, it does not seem to perform better than all comparing strategies because CCS initialize content's validities to perform caching operations and for each content the validity is cached at all the nodes along the delivery path. Therefore, a large number of requests need to traverse several hops to check validity at publisher node. Thus, the link utilization is increased. However, the TCS achieves better performance with respect to the cache size and popularity parameters in term of reduced link congestion by caching the popular content at network edges. Furthermore, the PCS performs slightly worse than TCS because, PCS caches the content at the edge node and centrality node in an AS. As a result, path becomes congested because several requests cannot fulfill at centrality node and needs to forward to the edge node to retrieve the desired content. As a result, the network link utilization (link load) is increased.

The link load in PUC is determined by the CHs because when the CH is selected, the content is cached only at that CH and multiple users' requests for diverse contents must traverse

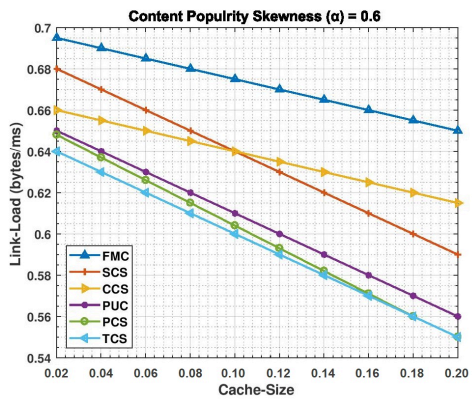
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal



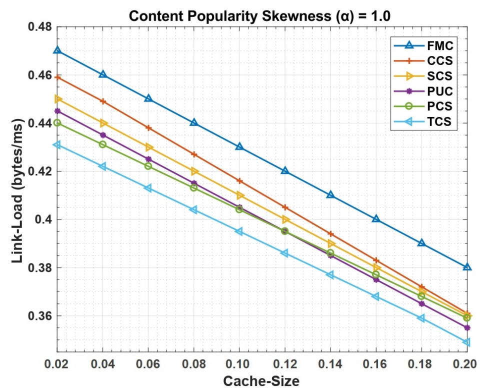
(a) Link load measure on GEANT topology



(b) Link load measure on GEANT topology



(c) Link load measure on WIDE topology



(d) Link load measure on WIDE topology

Fig. 11: Link load measure on different topologies for different popularity models.

several hops to find the content from the original publishers. As a result, the path becomes congested, and performance suffers.

VI. SUMMARY AND INSIGHTS

Currently, IoT-based networks are built on the TCP/IP Internet architecture, which was designed to connect a large number of computers using limited address spaces to share limited network resources. Indeed, it was not intended to meet the requirements of modern IoT-based networks. Additional needs for IoT situations include scalability of diverse devices and data transmission across the underlying infrastructure. Usually, heterogeneous devices such as watches, computers, vehicles, and mobile phones are involved to make an IoT-based network [4], [68]. The connected users can generate and retrieve various types of content via these smart devices. To meet the requirements of IoT scenarios, the ICN paradigm is regarded as an ideal candidate because it offers a number of promising features such as scalability, naming-based data forwarding, in-network caching, flexible mobility management, and content level security. In fact, all of these characteristics are clearly appropriate for IoT-based applications. Furthermore, in-network caching is a remarkable feature of ICN architectures that can handle issues related to information dissemination by caching the contents at intermediate network nodes and can reduce content retrieval latency. Moreover, the naming feature has the ability to detach the end users from the server and eliminate the address-space scarcity related problems of TCP/IP. It provides efficient scalability and offer flexible data retrieval produced by IoT-based applications. In line with this assertion, in this study, an NDN-based caching strategy for IoT such as FMC, SCS, CCS, PCS, and TCS are chosen and those strategies are comprehensively studied to find an efficient caching strategy. The performance analysis is carried out in terms of latency, hit ratio, stretch, and link load.

From our comparative analysis, we have observed that CCS aims to improve the validities of contents and achieves better performance in terms of content retrieval latency. The reason is that, in CCS the contents' validities are observed and measured whenever a request is received at any of the network node. In fact, a large number of requests are received at a network and for each request, the validity of corresponding content is computed to check the content's popularity. Consequently, it takes extra time to respond the users and hence, the latency increases. Similarly, CCS performs poor to achieve better cache hit rate because it considers validity of a content to perform caching operations that takes extra time to calculate popularity of a content. Likewise, when the cache of betweenness centrality node becomes full, a number of incoming requests need to be forwarded to the main source of data that increases the overall path stretch. Sometime, the betweenness centrality node delivers high performance, while it results in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

congestion as the cache of centrality node becomes full. The reason is that, if a content is most popular and its validity at caching node becomes higher as compared to its original validity at the publisher, the content will be deleted from the intermediate node. Therefore, a large number of subsequent requests need to be forwarded via several hops the publisher node. As a result, the overall caching performance is reduced.

The objective of FMC is to maximize the caching efficiency at the edges (subscriber's nodes). However, it increases the communication overhead because in FMC, the UAG sends maximum number of data packets to the caching nodes. In our evaluation we noticed that, FMC achieves better performance of latency as compared to CCS because it provides share caching system and the contents are cached near the end-user that reduces the latency and path stretch. However, it sends the maximum data packets to the caching node that increases the link overhead.

We observed that SCS performs better to some extant as compared to FMC because, SCS caches the popular contents at multiple deputy nodes and when caching node is in sleeping mode, the content is accessed from the other deputy node(s). Consequently, a smaller number of requests may be sent to the main publisher node and thus, the overall link load, path stretch gets reduced and cache hit is improved.

TCS also achieves better performance in terms of cache hit rate and link load by caching the content at edge nodes. However, if a popular content is cached at edge node, the subsequent requests from other nodes cannot access that content. As a result, the incoming requests from other edge nodes need to traverse several hops to retrieve the same content from the publisher node. Hence, the overall stretch performance decreased. The PCS performs better in term of content retrieval latency, cache hit ratio, and stretch ratio. The reason is that, the PCS implements the centrality-based caching without affecting the contents' popularity. However, PCS performs poor in terms of link load poor because PCS caches the content at the edge node and centrality node in an AS. Therefore, path becomes congested and most of requests cannot be fulfilled at centrality node and needs be to forward to the edge node to retrieve the desired content. Thus, the network link utilization (link load) is increases.

The PUC improves the network performance by caching the content at nearest CH. Therefore, a large number of subsequent requests are accomplished from CH nodes. In addition, the PUC show higher cache hit performance because it reduces the distance by caching the content at CH to fulfill the demands of subsequent users' requests. Moreover, PUC reduces the stretch between end users and the cached copy of content at CH. Therefore, it shows better performance in terms of stretch. However, in PUC, the link load is determined by the CHs because when a CH is selected, the content is cached only at that CH and multiple users' requests for diverse contents must traverse several hops to reach the original publishers. Therefore, a large number of requests and corresponding contents are transmitted over additional hops to fulfill the end-users' demands. As a result, the path becomes congested and the performance suffers.

Table III shows the summary of comparative analysis of PUC, PCS, TCS, SCS, FMC and CCS.

TABLE III: Performance Analysis of PUC, PCS, TCS, SCS, FMS, CCS.

Performance Metrics	PUC	PCS	TCS	SCS	FMC	CCS
Low latency	✓	✓	✓	✗	✗	✗
Moderate latency	✗	✗	✗	✗	✓	✓
High latency	✗	✗	✗	✓	✗	✗
Low cache hit ratio	✗	✗	✗	✓	✗	✗
Moderate cache hit ratio	✗	✗	✗	✗	✓	✗
High cache hit ratio	✓	✓	✓	✗	✗	✓
Short path stretch	✓	✓	✗	✗	✗	✗
Moderate path stretch	✗	✗	✗	✗	✓	✗
Long path stretch	✗	✗	✗	✓	✗	✓
Lower link load	✗	✗	✓	✗	✗	✗
Moderate link load	✓	✓	✗	✓	✗	✓
High link load	✗	✗	✗	✗	✓	✗

VII. FUTURE DIRECTIONS AND OPEN RESEARCH CHALLENGES

Caching is a very useful feature of ICN architectures for increasing data availability, reducing latency, and improving spectral and energy efficiency. However, designing an effective caching model in general is not a trivial task. The application requirements, heterogenous networks, resource limitation (particularly in IoT case), and other factors pose extra challenges, and therefore, several research problems are open and need further heed. This paper reports on the analyses of the recent NDN-based caching models for IoT environment–but this is only tip of the iceberg of the opportunities it makes available. What follows are a few suggestions for research questions that other researchers may want to explore.

A. Content Caching Decission

The cache everything everywhere approach is not useful, resulting in the wastage of cache resources and creating a considerable amount of redundant data in the network. Data replacement policies that will behave according to the behavior of the content and interest are needed. Caching in general creates three main questions: ***What to cache?*** It is useless to cache all the content in the network. Therefore, it is necessary to consider the popularity of content and determine what content to cache. Many users request the same content, and therefore, unpopular content has a negative impact on the utilization of caching. ***How to cache?*** It is necessary to evaluate the reputation of content rather the applying traditional caching policies, such as the least recently used (LRU), least frequently used (LFU), and first in first out (FIFO). ***Where to cache?*** As in ICN the cache is distributed in nature, it creates another challenge for the deployment of caching. Meaning that where to cache the content: at the device, base station, or at the core network. Partitioning cache at various location needs further attention. Moreover, in IoT environment, the storage capability of IoT devices is limited and therefore, it is very important to decide what content and how long the content will be cached, and which content is needed to be evicted when the cache fills up to accommodate the new content [69], [70].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

B. Popularity of Content

Measuring the popularity of incoming request for content plays a vital role to enhance the caching performance. However, it comes with several challenges such as dynamic content and fake popularity index by malicious users. In literature mostly a threshold value is used based on the historical information of the content requests [66], [71]. However, the threshold value may change over time particularly for the case of dynamic content and use of constant threshold value may decrease the performance of the caching systems. Moreover, it is possible that a malicious user can inject fake information to show that the content is popular and can make the cache system ineffective. Researchers should explore more mechanism that take into account various factors of caching popularity such as dynamic nature of content, and who can make the popularity index.

To this end, there could be multiple ways around to optimize the threshold value; either by running extensive simulation experiments, or by analyzing the traffic on real testbed systems. Researchers are encouraged to think about detailed simulation and testbed analysis. Moreover, various machine learning algorithms, statistical modeling techniques such as regression analysis etc. can be adopted to learn the system behavior to make the caching decision adaptive.

C. CacheFreshness

Although caching is considered as the most beneficial feature to enhance the overall network performance, the freshness is a critical challenge particularly in IoT environment [72]. In IoT environment, the contents are transient in nature and needed to be updated frequently by the producers that impose rigorous requirements in terms of data freshness. All such updated information can be received through specifying freshness value. Due to such transient nature of IoT content, the freshness value in ICN-based caching is forced to be included as the caching operation is performed. Therefore, caching strategies dealing with freshness are highly important for ICN-based IoTs. However, in ICN-based caching no efficient mechanisms are available, and the researchers are encouraged to propose more efficient and robust cache freshness mechanism for better caching performance in ICN-based IoT scenarios.

D. Cache Coherency

In cache coherency, the validity of contents in the cache is checked. As ICN in-network caching is distributed in nature, cache coherency becomes really important to check reliability of shared data cached at multiple locations. However, in ICN-based IoT, it is difficult to make the cache coherent since a locally cached copy of a content can be outdated, as a result, ICN cache coherence can largely impact the IoT-based scenarios. The reason is that, in IoT, the current status of the network can be changed at any time and there is need to check the state of cached content validity regularly. Moreover, there is lack of efficient cache coherence protocols. To the best of our knowledge, there is only one cache coherency protocol for ICN-based IoTs [1]. In order to provide content validation in IoT applications, we strongly encourage researchers to work on ICN-based coherency protocols.

E. Cache Threats

Caching alleviates congestion and latency problems over the busy Internet, especially for popular content, like video streaming. In the traditional Internet (i.e., IP based Internet) there is well known caching threats. However, in ICN/NDN architectures these threats are worrisome. The reason is that the malicious network nodes such as publishers and routers can announce, update, and distribute malicious data that could be very difficult to detect over the Internet. On one hand the in-network caching feature is very helpful to handle the issues related to the information dissemination by caching the contents at intermediate network nodes and has the ability to reduce content retrieval latency. On the other hand, it creates well know cache challenges such as cache snooping, cache pollution, and cache poisoning. In cache snooping the malicious users analyze the content stored in caches and predict the user behavior on the Internet. Whereas in cache pollution the malicious users fill caches with unpopular or irrelevant content rather than popular content and making caches ineffective for legitimate users. In cache poisoning the caches are filled with illegal, fake, or forged content and this kind of threat also make the cache ineffective for the legitimate users.

Moreover, due to transparent nature of in-network content caching poses security issues for producers. The reason is that in ICN, any node can cache the data even if it is not desirable [73]. This creates trust issues in ICN/NDN and ensuring trusted data and avoiding caches that provide bogus data are promising research directions. Defining trust relationships between data producers, content stores, and content requesters is a direction that requires further investigation..

F. Energy Optimization

Energy consumption is one of the fundamental issues of IoT-based applications particularly in resource constrained environment. To this end, caching can play a vital role in reducing energy consumption. As caching reduces the number of hops traverse by a user request to retrieve the data/content from the source, the energy usage can be minimized. The reason is that the distance between user and the caching node is small and the requested content is cached near the user [74]. The closer the content to user(s), the less energy consumption will be in the network and higher number of content requests can be successfully served from ICN caches via a cache hit, and the system can achieve high power saving. Researchers should explore further the correlation between power consumption and caching schemes taking into account the cache capacity for efficiently utilizing network resource according to network traffic and content popularity level. Such kind of energy efficient cache policies can considerably improve network performance, especially toward the goal of green Internet design. Moreover, we also plan is to extend this study as our future work by evaluating the energy consumption of NDN-based caching strategies for IoT.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3110977, IEEE Internet of Things Journal

G. Testbed Implementation

To date, many ICN/NDN solutions proposed in the surveyed literature are not deeply evaluated. Specifically caching models are mostly evaluated via simulators. Evaluating caching models on simulators may not be fair evaluations due to diversity in applications with varying in scale, resources, hardware, priorities, and mobility. As a result, various platform may generate different results. It is very important to come up with more realistic testbed evaluations in order to check the behavior of the proposed models. Many testbeds have been developed for ICN architectures [75]–[79] and they should be more extensively used to provide better knowledge about ICN caching solutions. In particular, solutions that hamper content caching or limit request rates may negatively affect the user experience and should be better investigated.

VIII. Conclusion

This article aims to provide a broad view and insights regarding NDN-based IoT caching on the network layer. To this end, an extensive comparative performance analysis of NDN-based caching strategies for IoT is presented. Specifically, PUC, PCS, TCS, SCS, CCS, and FMC strategies are selected for the performance analysis. For fair evaluation, a simulation environment was created using Icarus caching-based simulator which is specifically designed for the evaluation of NDN-based caching strategies. As a performance metrics, most well-known and relevant metrics such as content retrieval latency, cache hit ratio, stretch, and link load are considered. Simulation results show that the PCS and PUC have performed better to achieve high performance in terms of content retrieval latency, cache hit ratio, stretch, and link load as compared to the other caching strategies. TCS, on the other hand, outperforms PCS in terms of link load performance.

It has been discovered that NDN-based caching solutions can significantly increase performance in an IoT-based environment due to its in-networking caching capability, which affects content retrieval delay and reduces network resource utilisation. We anticipate that this research will contribute to a deeper understanding of present IP-based IoT difficulties as well as insights into new architectural designs and their deployment for the future Internet. Furthermore, this research could help with the deployment of ICN/NDN for emerging technologies such as edge/fog computing, software defined networking, and the Internet of Everything (IoE). These caching solutions are the most beneficial to end users because they reduce response latency, shorten the stretch path, reduce link load, and improve data availability with less delay.

References

[1] A. M. Alberti, M. A. F. Casaroli, D. Singh, and R. da Rosa Righi, "Naming and name resolution in the future internet: Introducing the novagenesis approach," *Future Generation Computer Systems*, vol. 67, pp. 163–179, 2017.

[2] F. M. Al-Turjman, M. Imran, and A. V. Vasilakos, "Value-based caching in information-centric wireless body area networks," *Sensors*, vol. 17, no. 1, p. 181, 2017.

[3] M. Alhowaidi, B. Ramamurthy, B. Bockelman, and D. Swanson, "Enhancing the sdtma-ndn architecture for transferring the scientific data software using named data networking," *Computer Networks*, vol. 166, p. 106954, 2020.

[4] M. Amadeo, C. Campolo, and A. Molinaro, "Information-centric networking for connected vehicles: a survey and future perspectives," *IEEE Communications Magazine*, vol. 54, no. 2, pp. 98–104, 2016.

[5] M. Amadeo, C. Campolo, A. Molinaro, and G. Ruggeri, "Content-centric wireless networking: A survey," *Computer Networks*, vol. 72, pp. 1–13, 2014.

[6] M. Arifuzzaman, Y. Keping, Q. N. Nguyen, and S. Takuro, "Locating the content in the locality: Icn caching and routing strategy revisited," in *2015 European Conference on Networks and Communications (EuCNC)*. IEEE, 2015, pp. 423–428.

[7] R. Ullah, M. A. U. Rehman, M. A. Naeem, B.-S. Kim, and S. Mastorakis, "Icn with edge for 5g: Exploiting in-network caching in icn-based edge computing for 5g networks," *Future Generation Computer Systems*, 2020.

[8] A. A. Barakabitze, T. Xiaoheng, and G. Tan, "A survey on naming, name resolution and data routing in information centric networking (icn)," *Int. J. Adv. Res. Comput. Commun. Eng*, vol. 3, no. 10, pp. 8322–8330, 2014.

[9] S. Arshad, M. A. Azam, M. H. Rehmani, and J. Loo, "Recent advances in information-centric networking-based internet of things (icn-iot)," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2128–2158, 2018.

[10] L. Atzori, A. Iera, and G. Morabito, "Understanding the internet of things: definition, potentials, and societal role of a fast evolving paradigm," *Ad Hoc Networks*, vol. 56, pp. 122–140, 2017.

[11] M. A. U. Rehman, R. Ullah, and B.-S. Kim, "Ninq: Name-integrated query framework for named-data networking of things," *Sensors*, vol. 19, no. 13, p. 2906, Jun 2019. [Online]. Available: http://dx.doi.org/10.3390/s19132906

[12] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[13] G. Acs, M. Conti, P. Gasti, C. Ghali, G. Tsudik, and C. A. Wood, "Privacy-aware caching in information-centric networking," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 2, pp. 313–328, 2017.

[14] M. Amadeo, C. Campolo, A. Molinaro, and G. Ruggeri, "Iot data processing at the edge with named data networking," in *European Wireless 2018; 24th European Wireless Conference*. VDE, 2018, pp. 1–6.

[15] O. Hahm, E. Baccelli, T. C. Schmidt, M. Wählisch, C. Adjih, and L. Massoulié, "Low-power internet of things with ndn & cooperative caching," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, 2017, pp. 98–108.

[16] P. Obba and I. K. Okakwu, "Performance evaluation of ipv6 and ipv4 for future technologies," in *Advances in Information and Communication: Proceedings of the 2020 Future of Information and Communication Conference (FICC).*, vol. 1. Springer Nature, 2020, p. 15.

[17] V. Sivaraman, H. H. Gharakheili, C. Fernandes, N. Clark, and T. Karliychuk, "Smart iot devices in the home: Security and privacy implications," *IEEE Technology and Society Magazine*, vol. 37, no. 2, pp. 71–79, 2018.

[18] M. Zhang, H. Luo, and H. Zhang, "A survey of caching mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1473–1499, 2015.

[19] L. Wei, Y. Chen, Y. Zhang, L. Zhao, and L. Chen, "Pspl: A generalized model to convert existing neighbor discovery algorithms to highly-efficient asymmetric ones for heterogeneous iot devices," *IEEE Internet of Things Journal*, 2020.

[20] R. Ullah, M. A. U. Rehman, and B. S. Kim, "Hierarchical name-based mechanism for push-data broadcast control in information-centric multihop wireless networks," *Sensors*, vol. 19, no. 14, p. 3034, 2019.

[21] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE communications surveys & tutorials*, vol. 16, no. 2, pp. 1024–1049, 2013.

[22] Y. Xu, Y. Li, S. Ci, T. Lin, and F. Chen, "Distributed caching via rewarding: An incentive caching model for icn," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[23] H. Wu, J. Li, J. Zhi, Y. Ren, and L. Li, "Design and evaluation of probabilistic caching in information-centric networking," *IEEE Access*, vol. 6, pp. 32 754–32 768, 2018.

[24] D. Saxena, V. Raychoudhury, N. Suri, C. Becker, and J. Cao, "Named data networking: a survey," *Computer Science Review*, vol. 19, pp. 15–55, 2016.

[25] S. J. Sheather and M. C. Jones, "A reliable data-based bandwidth selection method for kernel density estimation," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 53, no. 3, pp. 683–690, 1991.

[26] M. A. Naeem, R. Ali, M. Alazab, M. Yhui, and Y. B. Zikria, "Enabling the content dissemination through caching in the state-of-the-art sustainable information and communication technologies," *Sustainable Cities and Society*, p. 102291, 2020.

[27] M. A. M. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "On the performance of caching and forwarding in information-centric networking for the iot," in *International Conference on Wired/Wireless Internet Communication*. Springer, 2015, pp. 313–326.

[28] D. Gupta, S. Rani, S. H. Ahmed, and R. Hussain, "Caching policies in ndn-iot architecture," in *Integration of WSN and IoT for Smart Cities*. Springer, 2020, pp. 43–64.

[29] Y. Qin, W. Yang, and W. Liu, "A probability-based caching strategy with consistent hash in named data networking," in *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*. IEEE, 2018, pp. 67–72.

[30] S. Rahel, A. Jamali, and S. El Kafhali, "Energy-efficient on caching in named data networking: A survey," in *2017 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech)*. IEEE, 2017, pp. 1–8.

[31] Z. Zhang, C.-H. Lung, I. Lambadaris, and M. St-Hilaire, "Iot data lifetime-based cooperative caching scheme for icn-iot networks," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–7.

[32] C. Gündogan, P. Kietzmann, T. C. Schmidt, M. Lenders, H. Petersen, M. Wählisch, M. Frey, and F. Shzu-Juraschek, "Information-centric networking for the industrial iot," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, 2017, pp. 214–215.

[33] G. C. Polyzos and N. Fotiou, "Building a reliable internet of things using information-centric networking," *Journal of Reliable Intelligent Environments*, vol. 1, no. 1, pp. 47–58, 2015.

[34] A. Lindgren, F. B. Abdesslem, B. Ahlgren, O. Schelén, and A. M. Malik, "Design choices for the iot in information-centric networks," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2016, pp. 882–888.

[35] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, and K. Drira, "How to cache in icn-based iot environments?" in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*. IEEE, 2017, pp. 1117–1124.

[36] D. Mars, S. M. Gammar, A. Lahmadi, and L. A. Saidane, "Using information centric networking in internet of things: a survey," *Wireless Personal Communications*, vol. 105, no. 1, pp. 87–103, 2019.

[37] A. Djama, B. Djamaa, and M. R. Senouci, "Information-centric networking solutions for the internet of things: A systematic mapping review," *Computer Communications*, 2020.

[38] M. A. Naeem, M. A. U. Rehman, R. Ullah, and B.-S. Kim, "A comparative performance analysis of popularity-based caching strategies in named data networking," *IEEE Access*, vol. 8, pp. 50 057–50 077, 2020.

[39] S. Lee, I. Yeom, and D. Kim, "T-caching: enhancing feasibility of in-network caching in icn," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1486–1498, 2020.

[40] D. Man, Q. Lu, Y. Wang, Y. Wu, X. Du, and M. Guizani, "An adaptive cache management approach in icn with pre-filter queues," *Computer Communications*, vol. 153, pp. 250–263, 2020.

[41] M. A. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for internet of things (iot) security," *IEEE Communications Surveys & Tutorials*, 2020.

[42] F. Firouzi, B. Farahani, M. Weinberger, G. DePace, and F. S. Aliee, "Iot fundamentals: Definitions, architectures, challenges, and promises," in *Intelligent Internet of Things*. Springer, 2020, pp. 3–50.

[43] A. Araldo, D. Rossi, and F. Martignon, "Cost-aware caching: Caching more (costly items) for less (isps operational expenditures)," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1316–1330, 2015.

[44] J. Li, H. Wu, B. Liu, J. Lu, Y. Wang, X. Wang, Y. Zhang, and L. Dong, "Popularity-driven coordinated caching in named data networking," in *2012 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*. IEEE, 2012, pp. 15–26.

[45] Z. Li, J.-C. Point, S. Ciftci, O. Eker, G. Mauri, M. Savi, and G. Verticale, "Icn based shared caching in future converged fixed and mobile network," in *2015 IEEE 16th International Conference on High Performance Switching and Routing (HPSR)*. IEEE, 2015, pp. 1–6.

[46] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, and K. Drira, "Cache coherence in machine-to-machine information centric networks," in *2015 IEEE 40th Conference on Local Computer Networks (LCN)*. IEEE, 2015, pp. 430–433.

[47] M. Moeng, A. K. Jones, and R. G. Melhem, "Weighted-tuple: Fast and accurate synchronization for parallel architecture simulators," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 8, pp. 2462–2474, 2015.

[48] M. A. Naeem, R. Ali, B.-S. Kim, S. A. Nor, and S. Hassan, "A periodic caching strategy solution for the smart city in information-centric internet of things," *Sustainability*, vol. 10, no. 7, p. 2576, 2018.

[49] A. S. Nowacki, W. Zhao, and Y. Y. Palesch, "A surrogate-primary replacement algorithm for response-adaptive randomization in stroke clinical trials," *Statistical methods in medical research*, vol. 26, no. 3, pp. 1078–1092, 2017.

[50] Y. Song, H. Ma, and L. Liu, "Tccn: Tag-assisted content centric networking for internet of things," in *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2015, pp. 1–9.

[51] I. U. Din, S. Hassan, A. Almogren, F. Ayub, and M. Guizani, "Puc: Packet update caching for energy efficient iot-based information-centric networking," *Future Generation Computer Systems*, vol. 111, pp. 634–643, 2020.

[52] L. Saino, I. Psaras, and G. Pavlou, "Icarus: a caching simulator for information centric networking (icn)," in *Proceedings of the 7th International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTOOLS '14. ICST, Brussels, Belgium, Belgium: ICST, 2014.

[53] B. Banerjee, A. Seetharam, and C. Tellambura, "Greedy caching: A latency-aware caching strategy for information-centric networks," in *2017 IFIP Networking Conference (IFIP Networking) and Workshops*. IEEE, 2017, pp. 1–9.

[54] A. S. Gill, L. D'Acunto, K. Trichias, and R. van Brandenburg, "Bid-cache: Auction-based in-network caching in icn," in *2016 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2016, pp. 1–6.

[55] A. Seetharam, "On caching and routing in information-centric networks," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 204–209, 2017.

[56] F. Kuliesius and P. Paulauskas, "Simulation of content caching in information centric networking," in *2015 Seventh International Conference on Ubiquitous and Future Networks*. IEEE, 2015, pp. 391–394.

[57] "Cisco annual internet report (2018-2023) white paper," *Available online: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (accessed on 15 March 2020)*, 2020.

[58] S. Khodaparas, S. Yousefi, and A. Benslimane, "A multi criteria cooperative caching scheme for internet of things," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[59] M. Meeker and L. Wu, "Internet trends 2018," 2018.

[60] B. Banerjee, A. Kulkarni, and A. Seetharam, "Greedy caching: An optimized content placement strategy for information-centric networks," *Computer Networks*, vol. 140, pp. 78–91, 2018.

[61] A. Banerjee, B. Banerjee, A. Seetharam, and C. Tellambura, "Content search and routing under custodian unavailability in information-centric networks," *Computer Networks*, vol. 141, pp. 92–101, 2018.

[62] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable icn," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 147–158, 2013.

[63] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 1–14.

[64] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012.

[65] M. A. Naeem, S. A. Nor, S. Hassan, and B.-S. Kim, "Performances of probabilistic caching strategies in content centric networking," *IEEE access*, vol. 6, pp. 58 807–58 825, 2018.

[66] Y. Meng, M. A. Naeem, R. Ali, and B.-S. Kim, "Ehcp: An efficient hybrid content placement strategy in named data network caching," *IEEE Access*, vol. 7, pp. 155 601–155 611, 2019.

[67] N. Abani, T. Braun, and M. Gerla, "Proactive caching with mobility prediction under uncertainty in information-centric networks," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, 2017, pp. 88–97.

[68] O. Ascigil, S. Reñé, G. Xylomenos, I. Psaras, and G. Pavlou, "A keyword-based icn-iot platform," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, 2017, pp. 22–28.

[69] S. Tarnoi, K. Suksomboon, W. Kumwilaisak, and Y. Ji, "Performance of probabilistic caching and cache replacement policies for content-centric networks," in *39th Annual IEEE Conference on Local Computer Networks*. IEEE, 2014, pp. 99–106.

[70] M. A. Naeem, S. A. Nor, S. Hassan, and B.-S. Kim, "Compound popular content caching strategy in named data networking," *Electronics*, vol. 8, no. 7, p. 771, 2019.

[71] Y. Meng, M. A. Naeem, R. Ali, Y. B. Zikria, and S. W. Kim, "Dcs: Distributed caching strategy at the edge of vehicular sensor networks in information-centric networking," *Sensors*, vol. 19, no. 20, p. 4407, 2019.

[72] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, K. Drira, and S. AlAhmadi, "Cache freshness in named data networking for the internet of things," *The Computer Journal*, vol. 61, no. 10, pp. 1496–1511, 2018.

[73] E. Mannes and C. Maziero, "Naming content on the network layer: a security analysis of the information-centric network model," *ACM Computing Surveys (CSUR)*, vol. 52, no. 3, pp. 1–28, 2019.

[74] R. Ullah, M. K. Afzal, and B.-S. Kim, "A novel parallel processing mechanism for data transmission in wireless content-centric networking," *Journal of Intelligent & Fuzzy Systems*, vol. 35, no. 6, pp. 5815–5825, 2018.

[75] R. Ullah, M. A. U. Rehman, and B.-S. Kim, "Design and implementation of an open source framework and prototype for named data networking-based edge cloud computing system," *IEEE Access*, vol. 7, pp. 57 741–57 759, 2019.

[76] R. Ullah, M. A. U. Rehman, and B. S. Kim, "Poster: A testbed implementation of ndn-based edge computing for mobile augmented reality," in *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications*, 2019, pp. 181–181.

[77] P. Kumar and S. Reddy, "Lessons learned from the deployment of testbed for precision agriculture," in *Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur-India*, 2019.

[78] P. H. V. Guimaraes, L. H. G. Ferraz, J. V. Torres, D. M. Mattos, I. D. Alvarenga, C. S. Rodrigues, O. C. M. Duarte *et al.*, "Experimenting content-centric networks in the future internet testbed environment," in *2013 IEEE International Conference on Communications Workshops (ICC)*. IEEE, 2013, pp. 1383–1387.

[79] H. Lim, A. Ni, D. Kim, Y.-B. Ko, S. Shannigrahi, and C. Papadopoulos, "Ndn construction for big science: Lessons learned from establishing a testbed," *IEEE Network*, vol. 32, no. 6, pp. 124–136, 2018.

Muhamad Ali Naeem received the M.Sc. degree in computer science from the COMSATS Institute of Information Technology, Pakistan, and the Ph.D. degree from the School of Computing, College of Arts and Sciences, Universiti Utara Malaysia. He is currently working with the School of Science, Guangdong University of Petrochemical Technology, Maoming, China. His major interests are in the field of information centric wireless networks, named data networking, and the Internet of Things.

Rehmat Ullah received his Ph.D. degree in electronics and computer engineering from Hongik University, South Korea, in February 2020. Currently, he is working as a post-doctoral research fellow with the School of Electronics, Electrical Engineering and Computer Science, at Queen's University Belfast, UK. Prior to joining Queen's, he was an assistant professor with the Department of Computer Engineering at Gachon University, South Korea. His research interests are in edge computing, information centric networking and 5G evolution and beyond with a recent focus on federated learning for edge computing systems. More information is available from www.rehmatkhan.com

Yahui Meng received the B.Eng degree in computer science and technology from Air Force Engineering University, and the M.Sc degree in software engineering from Huazhong University of Science and Technology. He is currently pursuing the Ph.D degree in computer science. He is a Ph.D Scholar at the InterNetWorks Research Lab, School of Computing, Universiti Utara Malaysia. Meanwhile, he is also currently a lecturer in School of Science, Guangdong University of Petrochemical Technology. His research interests included computer network security, communication, and transport protocols.

Rashid Ali [S17, M20] is currently an assistant professor at the School of Intelligent Mechatronics Engineering, Sejong University, Seoul, Korea. He received his Ph.D. degree (2019) in information and communication engineering from the Department of Information and Communication Engineering, Yeungnam University, Korea. His research interests include next-generation wireless networks, 5G and beyond, and reinforcement learning.

Bilal Ahmed Lodhi received the B.S. degree in Computer Science from Baqai University, Karachi, Pakistan, in 2004, an M.S. degree in Computer Science from the National University of Computer and Emerging Sciences (FAST), Islamabad, Pakistan, in 2009, and a Ph.D. degree in Computer Science from Korea University, Seoul, South Korea, in 2019. He worked as an ultrasound research engineer at Alpinion Medical Systems. From 2020, he is a Research Fellow at School of Electronics, Electrical Engineering and Computer Science (EEECS), Queen's University Belfast, United Kingdom. Prior to joining QUB, he was a Research Fellow at the University of Seoul (UoS) in Seoul, South Korea.