Received: 27 November 2021 | Accepted: 17 January 2022

DOI: 10.1049/smc2.12023

IET Smart Cities

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH

An optimal content indexing approach for named data networking in software-defined IoT system

Rajan Kumar Dudeja[1] | Amritpal Singh[2] | Rasmeet Singh Bali[1] | Gagangeet Singh Aujla[3]

[1]Computer Science & Engineering Department, Chandigarh University, Mohali, Punjab, India

[2]Chitkara University Institute of Engineering & Technology, Chitkara University, Rajpura, Punjab, India

[3]Department of Computer Science, Durham University, Durham, UK

Correspondence

Gagangeet Singh Aujla, Department of Computer Science, Durham University, Durham DH1 3LE, UK.
Email: gagangeet.s.aujla@durham.ac.uk

Funding information

Durham University

Abstract

The evolution of the Internet of Things (IoT) has increased the number of connected devices in the network. This has shifted the focus from IP-based network architecture towards content-centric networking (CCN). CCN eliminates the need for address-content binding in the conventional IP-based networks and allows the content to be accessed based on the name instead of the physical location. Named data networking (NDN) is a promising technique that can fulfil the increasing demand for connected devices through the CCN approach. NDN distributes the content on the network and focusses on the security of the content rather than the communication channel. However, the increase in traffic due to the escalation in the number of connected devices can lead to congestion in the network. The content distribution approach on the nodes is generalised and suitable for small networks. In the case of larger networks, an optimal approach is required to decide the optimal location to store the required content. However, a linear search approach is used to search (or lookup) the content in the assigned cache of the NDN node. In this work, the authors have combined the software-defined networking (SDN) with the NDN approach to overcome the above-highlighted challenge. Thus, the authors have designed an optimal content storage and indexing approach based on NDN-SDN coalesce in the IoT ecosystem. The proposed approach includes different phases, (a) a hashing-based content searching approach is formulated to reduce the look-up time of the content, (b) a red-black tree-based content storage approach is introduced for optimal utilisation of the assigned cache memory of the different NDN nodes, and (c) SDN controller facilitates automated network management and helps to administer the network requirements centrally and locate the content accordingly. The proposed approach was validated through the simulation experiments concerning network delay, packet rate, throughput, and cache hit ratio. The results obtained show the effectiveness of the proposed approach.

KEYWORDS

content-centric networking, forward information base, Internet of things, named data networking, pending interest table, red-black tree, software-defined networking

1 | INTRODUCTION

COVID-19 pandemic has shifted the focus of global organisations towards the new normal in the form of online applications and tools to run their businesses. According to a recent report [1], the one-third proportion of the small businesses has to shut down due to non-compliance with the new normal. Moreover, nearly 80% of the business organisations have shown their willingness to adopt online and digital tools to boost their businesses. A summary related to the adoption of digital tools and platforms is presented in Figure 1. In line with this, one of the most important revolutions of the era, namely

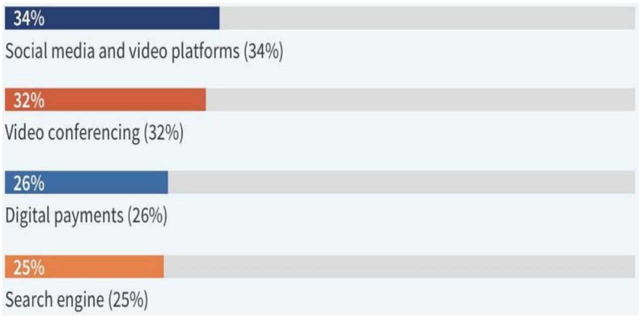

FIGURE 1 Increase in digital tools during pandemic situation [1]

the Internet of Things (IoT) has gained significance during these times. IoT devices are configured to monitor and transfer data from various digitised devices and platforms. However, the increasing dependence on IoT devices across the globe has increased the number of connected devices to the network in an exponential manner. The Internet services are bound to the network layer (IP addresses) to access the global connectivity. But, the limited IP addresses can bound the scalability of the networks and result in congestion in the network. Alternately, the distributed networks are more viable to provide the services to the end-users as compared to the standard networks due to their multi-facet benefits. However, the distributed network communication has to face several bottlenecks due to the point-to-point framework model. Therefore, an IP-independent approach is required to meet the demands of the connected users and overcome the above-discussed challenges.

Content-centric Networking (CCN) has emerged as an alternative to fulfil the demand of the connected world by allowing the content to be directly addressable and routable [2, 3]. *Named data networking* (NDN) is a promising approach that has flourished under the domain of CCN to meet the futuristic demands of the connected devices [4, 5]. NDN works according to the environmental behaviour of the network without relying on the IP addresses in the network. It is a receiver-driven communication platform that is independent of IP architecture. The producer and consumer are the key components in the NDN architecture. The **INTEREST** and **DATA** packets are transmitted in the network using the stateful forwarding plane. The **INTEREST** packet contains the information of the required content and the forwarders sent it in the network after keeping a state of the data request. The **INTEREST** packet is replaced by a **DATA** packet at the destination node and it is sent back to provide the requested data to the consumer. The forwarders erase the state after the corresponding **DATA** packet arrives back. The **DATA** packet follows the same path to reach the consumer as followed by the **INTEREST** packet.

1.1 | Research problem

The key features of the NDN, like location independence and network caching, make it popular and attractive. If in a case, the **INTEREST** packet is not available in the nearby node, then it is forwarded to the immediate node for content delivery. The same process continues till the requested data is not provided by the nodes. This approach is feasible for small networks as there are limited nodes and identification of the destination node to generate the **DATA** packet is not difficult. However, this may not be possible for a large network. The NDN approach is IP-independent; therefore, a single consumer can generate multiple **INTEREST** packets and the identification of the **DATA** packet is a tedious task and can create congestion in the network. In such a case, the configured network can freeze and therefore degrade the Quality of Service (QoS).

A content identification platform is required to locate the required data in the larger network without freezing the network. Moreover, NDN has to deal with a network packed with an array of names and the number of data that each user needs to interact with will be far higher than with the conventional TCP/IP architecture. Even more, the namespace is not limited and can be as large as it could get, thereby ending up in an unparalleled scale of the routing tables. Moreover, the search time relate to the **INTEREST** packet in the available cache of the NDN is also an important point to highlight. A linear search approach is used to search (or lookup) the content in the assigned cache of the NDN node. Therefore, an optimal content storage approach is also a key requirement to reduce the overall look-up time of the IM. Further, NDN needs a reliable partner like software-defined networking (SDN), which can help to track the data (or content) automatically and subsequently optimise the search. SDN decouples the network into data, control, and application planes to segregate the functionality of the configured network [6–8]. This helps the platform to behave flexibly and adapt as per the behaviour of the network. SDN also supports intelligence and can learn from previous experiences. This can help to move through the named data in a faster manner on each go. A centralised controller collects the details of the configured devices in the network and manages the network as per the defined policies to meet the service level agreement. The centralised controller in SDN can allow prioritisation of named data according to the need of the user. Using prioritisation, the controller can correlate the data and make it available locally. This way SDN can help NDN to locate the ***DATA*** packet in the larger networks without hampering the functionality of the network.

1.1.1 | Research questions

Keeping in view of the above-discussed research problem, there are several research questions (RQ) that need to be answered while realising the full potential of the proposed system.

- **RQ1:** How to develop an optimal content storage approach in the available cache memory of the NDN nodes?

- **RQ2:** How to develop an approach to reduce the overall look-up time of the content requested through the **INTEREST** packet on the NDN nodes?
- **RQ3:** How to decide the optimal gateway node for content relocation as per generated **INTEREST** packet?

1.2 | Research approach and key contributions

To answer the above research questions, we have designed a very systematic research approach as depicted in Figure 2. In this approach, the clusters are formulated on the basis of the similar Name-prefix approach to categorise the **INTEREST** packets. Then, a hashing-based content searching approach is formulated to reduce the look-up time of the content (Section 4.1). Further, a red-black tree-based content storage approach is introduced for optimal utilisation of the assigned cache memory of the different NDN nodes (Section 4.2). Afterwards, SDN is configured over the NDN network for appropriate gateway node selection for content relocation to reduce the overall throughput time to fetch the **DATA** packet (Section 4.3). Based on the above discussion, the main contributions of this article are highlighted below.

- A multi-layered SDN-NDN system model is proposed for cluster formulation according to the behaviour of the ***Interest*** packet for a similar type of traffic
- A software-defined directed network control model is designed to relocate the position of the requested data according to the behaviour of the nearby clusters to improve the QoS of the network
- An intelligent content storage/indexing approach is designed working in line with a software-defined directed network control model for IoT
- We simulate the proposed approach using the ndnSIM simulator for validation concerning the network delay, throughput, and packet rate

2 | RELATED WORK

Numerous solutions are provided by researchers considering the above-discussed challenges in IoT. For example, Wu et al. [9] introduced a scheme to integrate the IP with the NDN routing to improve the overall performance of the network. However, this work focussed on energy consumption rather than performance challenges. Yuan et al. [10] presented the performance evaluation of the NDN-based and HTTP-based content distribution schemes. The authors compared the simulated results with the wired networks and the HTTP-based approach in terms of latency and provided 10× greater throughput. But, this work does not consider the large-scale networks where the identification of **DATA** packets is challenging. Mauri et al. [11] highlighted the benefits of Information-Centric Networking architecture and discussed the content prefetching nature of NDN that can help users to retrieve the requested content in the vehicular scenario. The authors analysed the impact of the used topology on the content retrieval due to the mobility of the users and the size of the available caches. This work was specific to the vehicular scenario and its applicability in the IoT environment is not validated.

Yovita et al. [12] studied the change in the performance of the network due to the content storage size and requested interest packets by the users. The authors configured the Abilene network topology to improve the hit ratio. They validated the proposed approach based on the average delay and the cache hit ratio considering the number of interest packets and data packets stopped due to network congestion. In another work, Rezazad et al. [13] proposed a scheme to segregate the NDN traffic into various classes and planned to divide the stored content on the routers according to the defined classes. The authors used the Buffer Miss Equation to partition the storage space dynamically to improve the efficiency of the network. Similarly, Li et al. [14] introduced an advanced data structure approach by integrating the pending interest table (PIT) with the bloom filter to improve the overall performance of the network. However, none of the above-discussed proposals considered content or data prioritisation to locate data locally.

Yu et al. [15] proposed an in-network caching scheme to store the content according to the popularity rating on the routers to reduce the redundant content. The authors compared the results with the leave copies everywhere caching strategy to validate the proposed scheme. Ooaka et al. [16] proposed the encapsulation of NDN packets into the IP/UDP-based packets. The distinction between the NDN and IP-based packets is based on the different port numbers allocated to them. The proposed work enables the dual-mode working to handle both types of packets. However, the problem associated with these kinds of approaches is related to the hashing of different lengths of content-Name

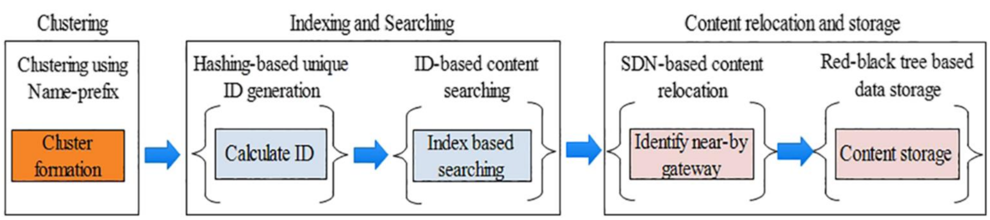

FIGURE 2 Schematic diagram for the research approach

in NDN to fix length bits in IP-based techniques. Also, these works have not utilised SDN for automatic control and optimised search.

Carvalho et al. [17] deployed the modified OpenFlow switches that can interact with NDN nodes and content servers to handle the network and storage-related queries. These switches are capable to query the data and content list from the server that provides them with a global view of the network. The content server is responsible for the location and storage duration of the content on various nodes. Although it enables the advantages of SDN with its centralised global view of the network, it brings hardware costs of changing the operating modes of switches. In another work, Mahmood et al. [18] proposed a state-full SDN approach that integrates with NDN. This approach employees SDN switch having capabilities to deal with NDN-related decisions during run-time without the central SDN server consultation. This reduces the communication cost among devices and improves the caching and other capabilities of NDN. Tantayakul et al. [19] proposed an approach for caching policy in NDN without modifying the user equipment. This approach integrates the caching policies with the SDN mobility scheme to improve the packet loss ratio. Here, the cache duration is decided on the basis of network quality assessment considering caching policies used in the approach are ON/OFF and adaptive caching techniques. Kalghoum et al. [20] discussed the methods to integrate SDN with a bloom filter-based data structure for the implementation of NDN networks. The authors suggest that this helps to improve the name and data searching capabilities of the SDN controller.

Table 1 provides a comparative analysis of various existing proposals. The above-discussed approaches have been validated for small-scale networks. However, the increase in the size of the network can also increase the complexity, and a lot of transformation can be seen in the devices used in the data plane. Thus, there is a constraint in the storage space of cache memory integrated on the routers. Hence, an approach is required to use the available cache memory optimally in order to increase the hit ratio and minimise the latency in the network. Moreover, the focus is also required in the direction to store the required content locally.

3 | NDN-SDN SYSTEM MODEL

We have proposed an SDN-NDN amalgamation in the form of a system model for the content indexing approach as shown in Figure 3. NDN is a content-centric networking approach used to transmit the content to the application layer directly. It works on a named data scheme and is independent of the IP addresses of various connected devices. The SDN-NDN amalgamation helps to identify the location of the content stored in the network. The various components in the context indexing approach are discussed below.

3.1 | NDN layer

NDN is a receiver-driven communication framework for content distribution on the configured devices in the network. It works on the basics of the on-demand delivery of data

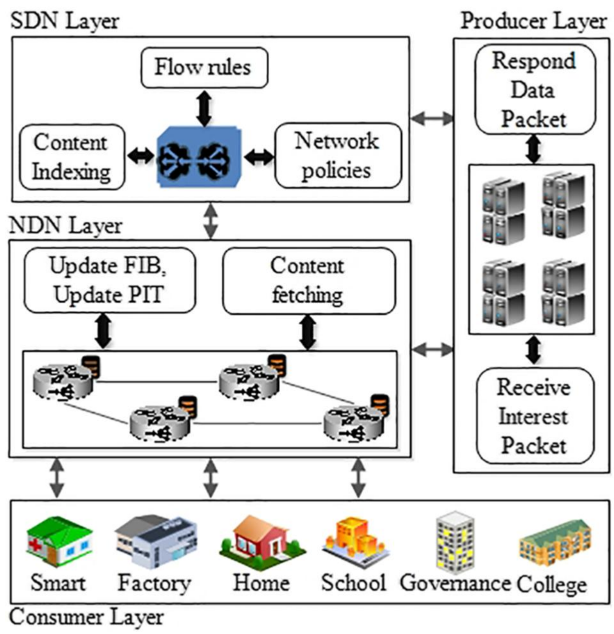

FIGURE 3 Proposed SDN-NDN system model. FIB, forward information base; NDN, named data networking; PIT, pending interest table; SDN, software-defined networking

TABLE 1 Analysis of existing proposals

Proposal	Description	1	3	4
Campolo et al. [4]	Incorporate NDN with edge computing framework	-	AI-based	NDN-Edge
Ayadi et al. [5]	Intelligent data forwarding scheme based on deep learning	Deep learning	-	NDN
Qin et al. [21]	Edge computing-aided congestion control scheme to improve the QoS	✓	✓	NDN-IoT-Edge
Kim et al. [22]	Proposed two cache decision factors to optimise the storage	-	Two cache	NDN
Rezazad et al. [23]	NDN traffic division into different classes for storage optimisation	-	Traffic based	NDN
Zhou et al. [24]	Mobility support architecture for content relocation	Centralised	-	NDN
Proposed	NDN-SDN framework for intelligent data relocation and searching	SDN based	Red-black tree	NDN-SDN

Note: 1: Content relocation approach, 2: Storage optimisation, 3: Cache management, 4: Environment.

Abbreviations: IoT, Internet of Things; NDN, named data networking; QoS, Quality of Service; SDN, software-defined networking.

packets and uses the conventional naming scheme to name the data generated by the NDN nodes installed in different organisations. The names of nodes are assigned based on their installed positions. Their positions define the *Name – Prefix* assigned to the data generated by them. The communication in the NDN framework is initialised at the receiver end, that is, the consumer through two packets: **INTEREST** and **DATA** packets. The complete content retrieval process as per the **INTEREST** packet is shown in Figure 4. All the configured nodes support the in-networking caching feature to store the data produced by the NDN producer node. It also maintains the data structure table, named: Forward information base (FIB) and PIT, used to maintain the record of all available nodes in the network and pending interest requests, respectively. PIT is used to store the pending interest and FIB keeps track of the remaining nodes in the configured network [25, 26].

3.2 | Consumer layer

The consumers can be located in different areas like organisations, institutions, or residential areas. They generate requests in the form of an **INTEREST** packet for the required data/service. The generated packet is forwarded to the connected gateway node to locate the destination node for specific data.

3.3 | Producer layer

The producer provides the content as per the **INTEREST** packet. The producer can be the configured routers in the network or the available data centres. According to the **INTEREST** packet, the producer creates a **DATA** packet in order to fulfil the requirements.

3.4 | SDN layer

SDN architecture is divided into three planes, namely, data, control, and application. The lower plane comprises various configured devices, like, switches, and routers. These network devices are further connected to the control plane using the southbound interface. The top layer provides the feedback to the control layer as per the requirement and accordingly the controller directs the configured network devices using the northbound interface to make the network more reactive and efficient. A centralised controller is used to handle the network traffic efficiently. The control plane communicates to the data plane using OpenFlow protocol standards to make the communication more generic. The SDN framework is a dynamic platform that can adapt its functionality according to the network policies. SDN controller collects the details of all the configured devices in the network and directs the devices as per the defined policies to fulfil the requirement of the end-user. SDN controls the network traffic by locating and storing the requested content on the nearby NDN nodes to fulfil the requirement of the consumer at the earliest.

4 | INTELLIGENT CONTENT STORAGE/INDEXING APPROACH

In this section, we propose a novel content storage/indexing approach for the selection of appropriate nodes and Red-black tree-based content storage and retrieval from the available caching memory. The proposed approach works on the SDN controller deployed over the NDN. The cluster formation for consumers is the foremost process of the proposed scheme. The cluster formation process is based on the requested services by different clusters (like organisations, factories, schools/colleges etc.). The required services from various nodes of organisations like medical services or educational institutes are forwarded to an individual cluster. The clustered consumer is connected with the NDN gateway for further processing. The gateway node is the source to connect the consumer with the other available nodes in the network. All the configured nodes in the network work under the supervision of the SDN controller [27]. The proposed scheme is dissected into three phases, hashing-based indexing, red-black-based storage management, and intelligent content relocation and retrieval approach

4.1 | Hashing-based indexing approach

In the NDN-based network, the requested content is distributed on the configured nodes on the network. The

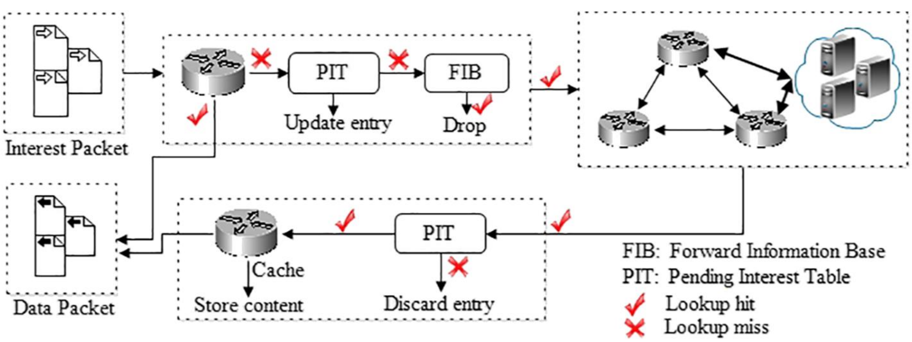

FIGURE 4 Packet forwarding process in named data networking. FIB, forward information base; PIT, pending interest table

conventional naming scheme is used to assign the name to the generated data (γ) from the producer (α). The demand-based sensors are deployed at various producers and each sensor is assigned a unique sensor id (β) for identification purposes. The nomenclature used to assign the name to the file is referred to $f(\alpha, \beta, \gamma)$. A conventional hashing approach is designed to assign a unique id to the generated file by the prouder to reduce the content look-up time. The hashing function to calculate a unique id is given below:

$$H_{(i \rightarrow \alpha, \beta, \gamma)} = \delta(\alpha \oplus \beta \oplus \gamma) \mod n \quad (1)$$

where H_i is the unique calculated integer value, n is the total number of files generated by various deployed sensors, and δ represent the ASCII value generated as per provided string.

4.2 | Red-black tree-based storage management

A Red-black is a binary data structure used to store the data at various blocks for efficient data searching. The representation of the red-black tree is highlighted in Figure 5. The red and black colouring scheme helped to organise the pieces of comparable data in the form of numbers or text. The nodes in the red-black tree can either carry keys or data and are called 'Internal nodes'. The top node in the tree is known as the root node and can be used as a reference node to insert the data at various nodes. The nodes underneath the root nodes are called child nodes and are responsible to carry the data. At the second level of the tree, the right child node value must be less as compared to the value of the left node in the tree. In the hierarchy of the red-black tree, a node can contain parent node, sibling node (25 and 10 as shown in Figure 5), and maybe a child node. During insertion of new data in the red-black tree, the nodes get re-arranged and re-painted to reduce the complexity in the worst cases. The following steps are used during re-arrange and re-colouring the nodes in the red-black tree.

- If the tree is empty, create a new node as the root node with colour coding black
- If the tree is not empty, create a new node as the child node and colour coding red

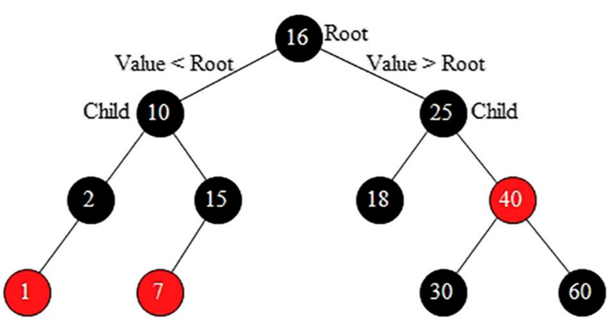

FIGURE 5 Red-black tree representation

- Further, check the colour code of the new node's parent node, exit if it is black
- In case, the new node's parent colour code is red, then check the colour code of the parent's sibling of the new node
  - If the colour code is black, then do suitable rotation and colouring [28]
  - In case, the colour coding is red, then re-colour it and further check either parent's parent of the new node is not the root node, then re-colour it and re-arrange the data in the nodes

4.3 | Intelligent content relocation and retrieval approach

The proposed scheme is receiver-oriented and it initiates the request message with the help of the consumer node for the required content. The requested message is initiated in the form of an **INTEREST** message (IM) to retrieve the required content with *Name – Prefix* nomenclature denoted by $C(\alpha, \beta, \gamma)$ to the immediately connected gateway. After receiving IM in the form of α, β, γ for a specific content, an entry in the PIT table is created. The PIT maintains the track of the pending requests for the generated content. Meanwhile, if the same content request is generated by any other node, the same entry will be updated in PIT. The configured NDN nodes are assigned with dedicated cache memory used to store the index table and content. The index table contains the unique ids according to the α, β, γ using Equation (1) and accordingly the content is stored in the cache memory using a red-black tree-based storage management system. The requested $C(\alpha, \beta, \gamma)$ message by the consumer is forwarded to the centralised controller to generate the hashing value (H_i) using Equation (1) and match the generated ID with the index table of the immediately connected NDN node to the consumer to fetch the requested content as shown in Figure 6 and forward the **DATA** packet to the concerned consumer for further processing.

In case, the generated id is not present in the index table of the immediately connected gateway node of the consumer, further according to the α, β, γ nomenclature, the *Packet_in* entry is created and forwarded to the SDN controller for further processing. NDN nodes forward the **INTEREST** packet to the next hope for content identification. After the identification of the required content from the producer, the SDN controller generates *Packet_out* entry and relocates the asked content from the destination node to the immediately connected gateway node to the consumer and update the entry as per the generated ID to the index table for further reference. In case, the cache memory of the gateway node is full, the standard cache replacement policies (like FIFO, LRU, and some hybrid page replacement techniques) are used to free the space of the respective node [29].

The main objective of the proposed scheme is to reduce the overhead of the network by locating the required content to the nearby gateway node and content look-up time. The cluster formation helps to generate similar requests by the consumers. The proposed scheme helps in the optimal

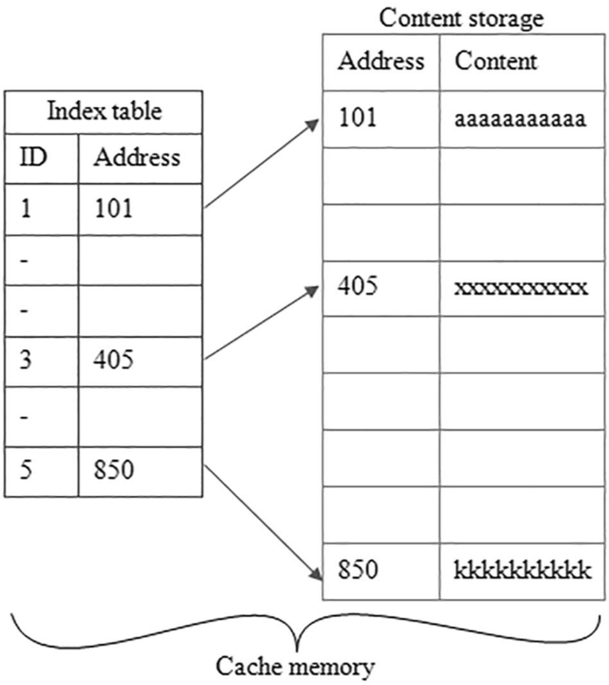

FIGURE 6 Index-content mapping in cache memory

utilisation of limited cache memory on the nodes and lower the number of look-up time. The step-by-step working of the proposed scheme is shown in Figure 7.

Algorithm 1 Intelligent content storage algorithm

```
INPUT: IM
OUTPUT: N(r)
Content Name-prefix: IM → C(α, β, γ)
Entry: C(α, β, γ) → PIT
while true do
    Push C(α, β, γ) into the respective cluster.
end while
Forward: C(α, β, γ) → SDN.
Calculate Hash ID using Equation (1).
for m = 1; m <= CN; m++ do
  for n = 1; n <= TR; n++ do
    while true do
      Match prefix for content: C(α, β, γ) ↔ N(r)
    end while
    Not Found: Generate Packet_IN
    Locate Name-prefix based producer
    Relocate the content to the nearest gateway
    Use Red-Black tree to save the content
    Return: N(r)
  end for
end for
```

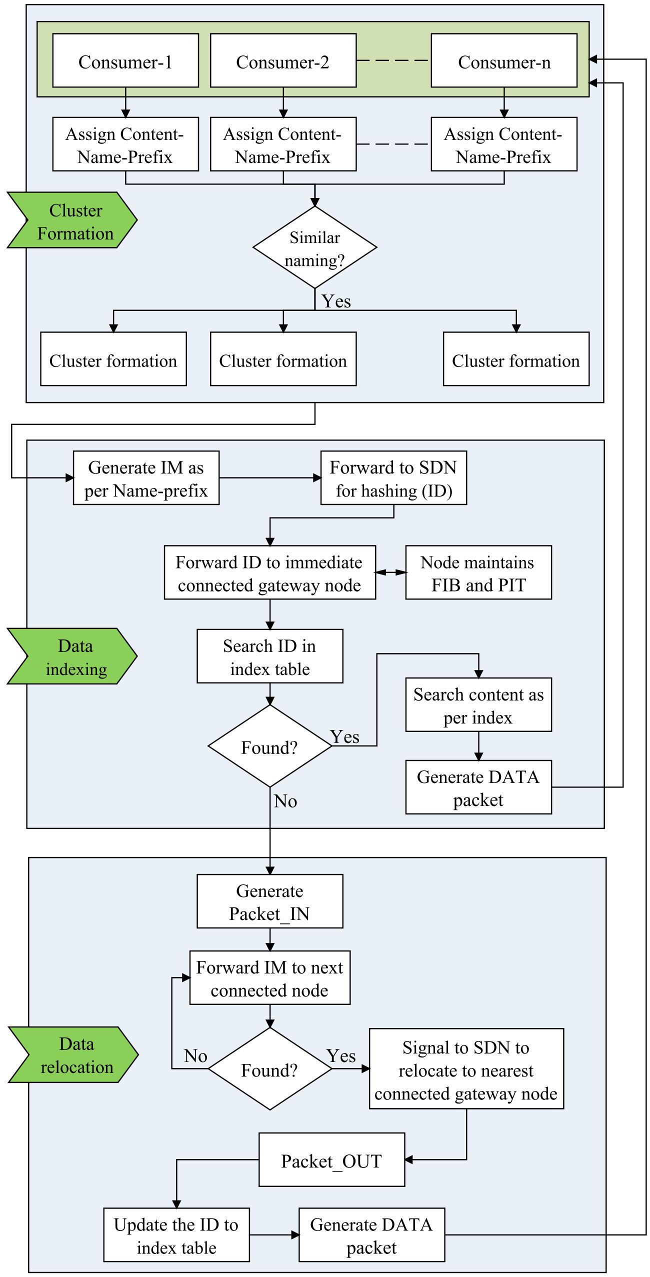

FIGURE 7 Flowchart of the intelligent content storage approach

Algorithm 1 has been formulated to depict the workflow of intelligent content storage. Here, **IM** is generated by the various connected sensors related to different services. The generated *Name-prefix* ($C(\alpha, \beta, \gamma)$) is mapped with the clusters and accordingly pushed into the respective cluster. The requested $C(\alpha, \beta, \gamma)$ is forwarded to the SDN controller to generate the ID using hashing function defined in Equation (1). Further, the total queued requests (TR) in each cluster (CN) are popped and matched with the available nodes in the network for the requested content. If not found, the *Packet_IN* message is forwarded to the SDN controller to generate a new flow rule and request the producer for the asked content

TABLE 2 Simulation parameters

Parameters	Description/Value
Number of nodes	100
Number of active nodes	30 (Approx.)
Number of consumers and producers	16 and 4
Number of routers	4
Topology used	Star
Routing method	Best-route
No. of interest packets generated	1000 s
Cache size	100
Virtual payload for data	1024 bytes
Link bandwidth	50 MB, 100 MB, 150 MB
Simulation time	50 min

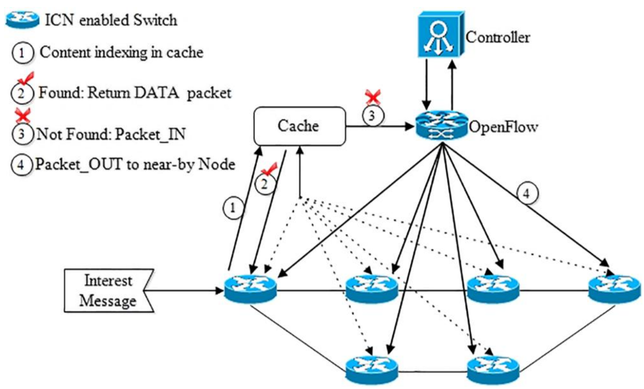

FIGURE 8 Scenario for named data networking over software-defined networking

and accordingly store the content on the nearby node ($N(r)$) using the Red-Black tree approach.

5 | RESULTS AND DISCUSSIONS

The proposed scheme is evaluated using the ndnSIM simulator [30] that performs basic NDN features and defines a node that acts as an SDN controller. Table 2 depicts the parameters used for simulation scenarios. These parameters define the number of active nodes at a given time, that is, approximately 30, and the simulation has been carried out for about 100 nodes. The numbers of producers, consumers, and NDN router nodes have been distributed in the ratio of 6:2:2, respectively. It also consists of one router node working as an SDN controller. The simulation was performed for a different number of nodes for their specific link bandwidths. The proposed approach performs its operations on real-time data generated by the producers. The data generated from the trace files (*cs trace, rate trace, app delay trace and drop trace*) were analysed to monitor the performance of the proposed approach. The performance evaluation is done on basis of standard evaluation parameters like delay, packet rate, and throughput.

Figure 8 represents the scenario of NDN over the SDN network. Just after an interest packet arrives at the first installed NDN switch in the network, it is forwarded to the assigned cache memory of the node using the OpenFlow protocol. In case, the content is found in the cache memory, the DATA packet is sent back to the NDN switch. In another way, the *Packet_IN* is generated and forwarded to the SDN controller for further processing and relocation of the content on the nearby gateway to minimise the look-up time.

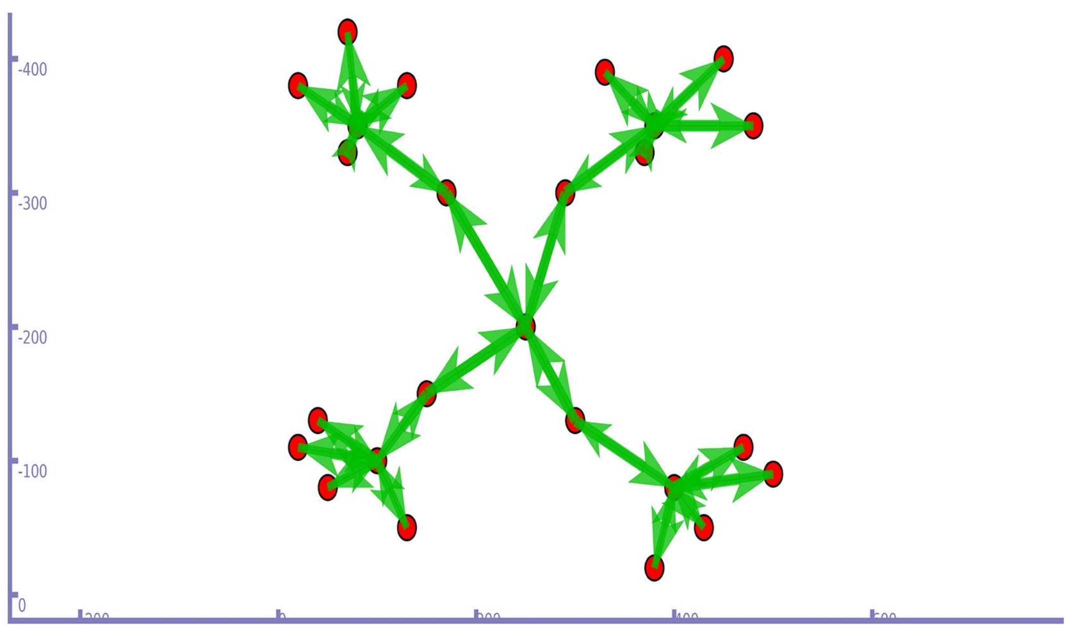

FIGURE 9 Topology of the proposed scenario

Figure 9 shows the simulation scenarios considered for the evaluation of the proposed approach. This scenario shows the partial representation of nodes acting as producers, consumers, NDN routers, and central SDN controller deployed as a star topology for the implementation. It shows that the nodes on outer edges known as consumer nodes form a cluster. The clusters are connected to NDN nodes acting as router nodes. These nodes are further connected to producer nodes. There is one router node (controller) at the centre that connects all the NDN nodes along with producer nodes.

Figure 10 shows the variation of packet delay concerning a variation in the sequence number of data packets generated during simulation for various bandwidth links. It shows that the initial delay in the network for bandwidths is higher but as the simulation progresses, the delay reaches its optimal level. After a certain period, it reaches a saturation level. The variation in delay for various bandwidths links is less than 0.1 μs, which shows the effectiveness of the proposed approach.

Figures 11 and 12 show the packet rate and throughput for various bandwidths as the function of simulation time, respectively. The behaviour of the network shows that initially, the number of packets and throughput are at lower levels but as the simulation progresses, the packet rate and throughput increase gradually and reach a peak value of ≈960 packets per sec and 625 kbps, respectively. After this, both values reach a saturation point. The above results show the effectiveness of the proposed approach for various bandwidth links concerning packet rate and throughput. Figure 13 compares the cache hit ratio of the SDN-enabled proposed approach with Standard cache techniques and optimal caching strategies. It shows that standard caching techniques LFU, LRU have a hit ratio between 20%–30%, whereas improved optimal techniques that stores data on a popularity basis shows a cache hit ratio of around 45%. The proposed SDN-based approach makes data available at gateway nodes near to consumer end. This improves the overall cache hit ratio to more than 50%. This shows the applicability of the SDN-based approach.

Figure 14 compares the CPU usage with a cache hit ratio. The CPU usage is checked with the TOP command in

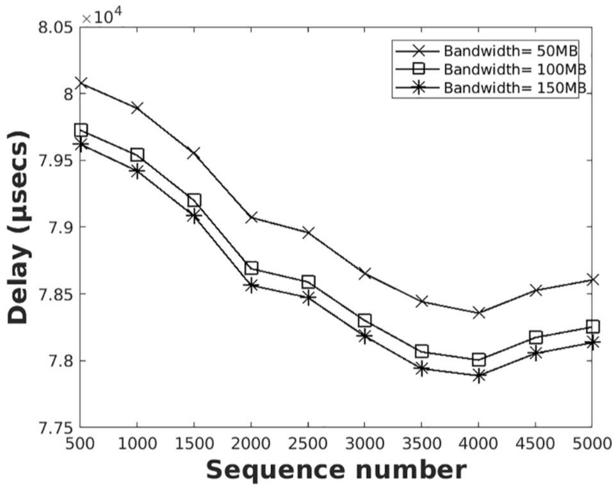

FIGURE 10 Avg delay v/s packet sequence number

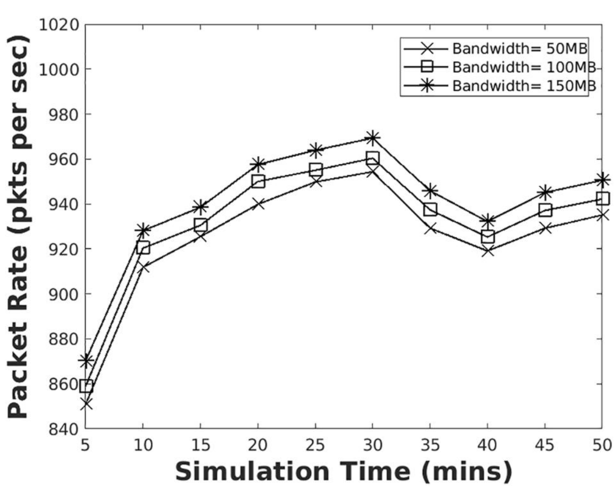

FIGURE 11 Packet rate v/s simulation time

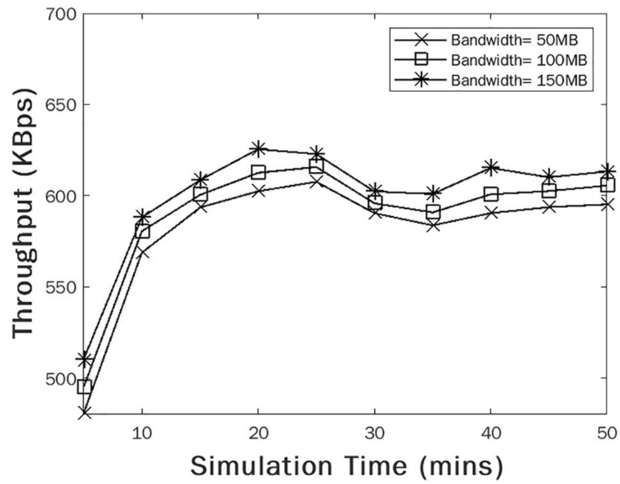

FIGURE 12 Throughput v/s simulation time

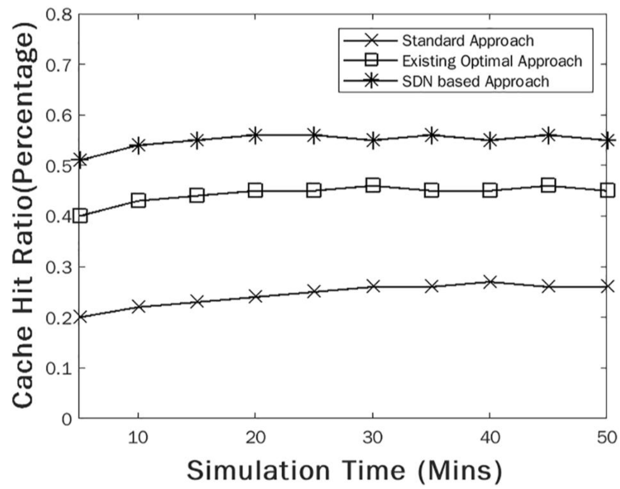

FIGURE 13 Cache hit ratio v/s simulation time. SDN, software-defined networking

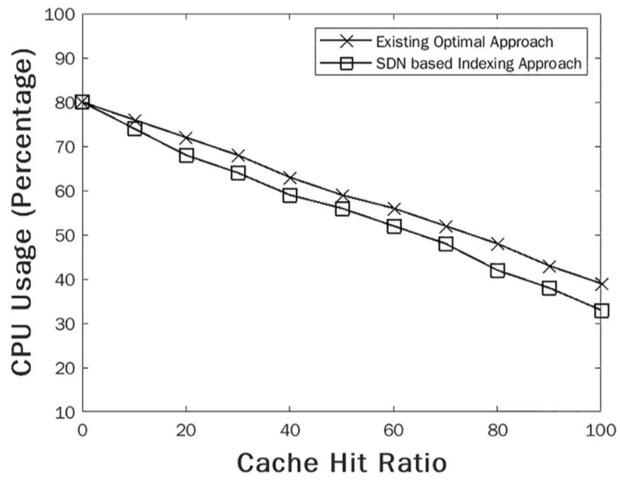

FIGURE 14 CPU percentage usage v/s cache hit ratio SDN, software-defined networking

UBUNTU. The total cache size of 5000 is projected for 100 nodes. The graph drawn for a cache hit ratio is from 0 to 1. It shows that with the increase in the cache hit ratio, the CPU usage is reduced. It starts with almost 80% and is projected to reach up to halve for the cache hit ratio of 1 [31]. The SDN-based indexing approach is projected to use around 33% of CPU for the cache hit ratio of 1. This is due to its efficient indexing-based storage mechanism of data in the cache of NDN nodes.

6 | CONCLUSION

In this work, we have proposed an optimal content storage/indexing approach working in NDN-SDN-based system architecture. This work tries to utilise the capabilities of SDN and NDN to create a novel coalesce to eliminate the bottlenecks of traditional IP-based networks. It provides ample space to attach everything to the Internet without the need for an IP address. As the traffic increases, NDN networks must handle the content delivery model optimally. Therefore, the proposed scheme integrates the SDN controller over the NDN network for efficient content delivery. SDN directs the NDN network to efficiently handle the data delivery model. The proposed NDN-SDN model has been validated using a simulated environment. The performance of the proposed approach was verified based on delay, packet rate, and throughput concerning different bandwidth links. The results obtained depict the effectiveness of the proposed approach in the considered setup.

ACKNOWLEDGEMENT

This work is partially supported through Startup Fund provided by the Durham University, UK.

CONFLICT OF INTEREST

No conflicts.

PERMISSION TO REPRODUCE MATERIALS FROM OTHER SOURCES

None.

DATA AVAILABILITY STATEMENT

No data involved.

ORCID

Gagangeet Singh Aujla https://orcid.org/0000-0002-2870-8938

REFERENCES

1. The covid-19 pandemic has challenged America's small business community. Digital tools are empowering survival today and recovery tomorrow. Digitally Empowered (2020). https://digitallyempowered.connectedcouncil.org/
2. Gulati, A., et al.: Adaptive caching for storage space analysis using content centric networking. In: 2018 IEEE Globecom Workshops (GC Wkshps). IEEE, pp. 1–6. (2018)
3. Gulati, A., et al.: Lattice-based secure and dependable data dissemination scheme for social internet of vehicles. IEEE Trans. Dependable Secure Comput. 18, 2520–2534 (2019)
4. Campolo, C., et al.: Towards named ai networking: unveiling the potential of ndn for edge ai. In: International Conference on Ad-Hoc Networks and Wireless, pp. 16–22. Springer (2020)
5. Ayadi, M.I., et al.: Deep learning in building management systems over ndn: use case of forwarding and HVAC control. In: 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 1192–1198. (2019)
6. Aujla, G.S., Singh, A., Kumar, N.: Adaptflow: adaptive flow forwarding scheme for software-defined industrial networks. IEEE Internet Things J. 7(7), 5843–5851 (2020)
7. Singh, A., Aujla, G.S., Bali, R.S.: Intent-based network for data dissemination in software-defined vehicular edge computing. IEEE Trans. Intell. Transport. Syst. 22, 5310–5318 (2020)
8. Singh, A., et al.: Deep-learning-based sdn model for internet of things: an incremental tensor train approach. IEEE Internet Things J. 7(7), 6302–6311 (2020)
9. Wu, T.Y., Wu, Y.W., Cheng, K.L.: An efficient ndn-based load adjustmentscheme for reduction of energy consumption. In: 2014 IEEE International Conference on Computer and Information Technology, pp. 533–538. (2014)
10. Yuan, H., Crowley, P.: Experimental evaluation of content distribution with ndn and http. In: 2013 Proceedings IEEE INFOCOM, pp. 240–244. (2013)
11. Mauri, G., et al.: Optimal content prefetching in ndn vehicle-to-infrastructure scenario. IEEE Trans. Veh. Technol. 66(3), 2513–2525 (2017)
12. Yovita, L.V., Syambas, N.R.: Content storage effect on the named data network traffic load. In: 2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA), pp. 1–5. (2017)
13. Rezazad, M., Tay, Y.C.: A cache miss equation for partitioning an ndn content store. In: Proceedings of the 9th Asian Internet Engineering Conference, ser. AINTEC '13. New York, NY, USA: Association for Computing Machinery, pp. 1–8. (2013). https://doi.org/10.1145/2534142.2534143
14. Li, Z., et al.: Mapit: an enhanced pending interest table for ndn with mapping bloom filter. IEEE Commun. Lett. 18(11), 1915–1918 (2014)
15. Yu, M., et al.: A caching strategy based on content popularity and router level for ndn. In: 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC), pp. 195–198. (2017)

16. Ooka, A., et al.: Openflow-based content-centric networking architecture and router implementation. In: 2013 Future Network & Mobile Summit. IEEE, pp. 1–10. (2013)
17. Carvalho, I., et al.: An introductory routing proposal for content centric networks using openflow. In: API, 7th Think-Tank Meeting, pp. 1–2. (2012)
18. Mahmood, A., et al.: Efficient caching through stateful sdn in named data networking. Trans. Emerg. Telecommun. Technol. 29(1), e3271 (2018)
19. Tantayakul, K., Dhaou, R., Paillassa, B.: Mobility management with caching policy over sdn architecture. In: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). IEEE, pp. 1–7. (2017)
20. Kalghoum, A., Saidane, L.A.: Fcr-ns: a novel caching and forwarding strategy for named data networking based on software defined networking. Cluster Comput. 22(3), 981–994. (2019)
21. Qin, J., et al.: Edge computing aided congestion control using neuro-dynamic programming in ndn. In: IEEE Global Communications Conference, pp. 1–6. (2020)
22. Kim, D., Lee, J.: An ndn cache management for mec. Appl. Sci. 10(3), 896 (2020)
23. Rezazad, M., Tay, Y.: A cache miss equation for partitioning an ndn content store. In: Proceedings of the 9th Asian Internet Engineering Conference, pp. 1–8. (2013)
24. Zhou, Z., et al.: Mobindn: a mobility support architecture for ndn. In: Proceedings of the 33rd Chinese Control Conference. IEEE, pp. 5515–5520. (2014)
25. Zhang, L., et al.: Named data networking. Comput. Commun. Rev. 44(3), 66–73 (2014)
26. Zhang, Z., et al.: An overview of security support in named data networking. IEEE Commun. Mag. 56(11), 62–68. (2018)
27. Yan, H., Gao, D., Su, W.: A hierarchical cluster-based caching for named data networking. In: 2016 IEEE/CIC International Conference on Communications in China (ICCC). IEEE, pp. 1–6. (2016)
28. Bhukya, D.P., Ramachandram, S.: Decision tree induction: an approach for data classification using avl-tree. Int. J. Electr. Comput. Eng. 2(4), 660 (2010)
29. Amadeo, M., et al.: Understanding name-based forwarding rules in software-defined named data networking. In: ICC 2020-2020 IEEE International Conference on Communications (ICC). IEEE, pp. 1–6. (2020)
30. Afanasyev, A., et al.: ndnsim: Ndn simulator for ns-3 (2012)
31. Kim, D., Kim, Y.: Enhancing ndn feasibility via dedicated routing and caching. Comput. Network. 126, 218–228 (2017)

How to cite this article: Dudeja, R.K., et al.: An optimal content indexing approach for named data networking in software-defined IoT system. IET Smart Cities. 4(1), 36–46 (2022). https://doi.org/10.1049/smc2.12023