Hindawi
Journal of Computer Networks and Communications
Volume 2021, Article ID 6640511, 16 pages
https://doi.org/10.1155/2021/6640511

Research Article

Pro^{NDN}: MCDM-Based Interest Forwarding and Cooperative Data Caching for Named Data Networking

Cong Pu

Department of Computer Sciences and Electrical Engineering, Marshall University, Huntington, WV 25755, USA

Correspondence should be addressed to Cong Pu; puc@marshall.edu

Received 8 November 2020; Revised 12 February 2021; Accepted 19 February 2021; Published 18 March 2021

Academic Editor: Salvatore Monteleone

Named data networking (NDN), as a specific architecture design of information-centric networking (ICN), has quickly became a promising candidate for future Internet architecture, where communications are driven by data names instead of IP addresses. To realize the NDN communication paradigm in the future Internet, two important features, stateful forwarding and in-network caching, have been proposed to cope with drawbacks of host-based communication protocols. The stateful forwarding is designed to maintain the state of pending Interest packets to guide Data packets back to requesting consumers, while the in-network caching is used to reduce both network traffic and data access delay to improve the overall performance of data access. However, the conventional stateful forwarding approach is not adaptive and responsive to diverse network conditions because it fails to consider multiple network metrics to make Interest forwarding decision. In addition, the default in-network caching strategy relies on storing each received Data packet regardless of various caching constraints and criteria, which causes the routers in the vicinity of data producers to suffer from excessive caching overhead. In this paper, we propose the Pro^{NDN}, a novel stateful forwarding and in-network caching strategy for NDN networks. The Pro^{NDN} consists of multicriteria decision-making (MCDM) based interest forwarding and cooperative data caching. The basic idea of the MCDM-based interest forwarding is to employ Technique for Order Performance by Similarity to Idea Solution (TOPSIS) to dynamically evaluate outgoing interface alternatives based on multiple network metrics and objectively select an optimal outgoing interface to forward the Interest packet. In addition, the cooperative data caching consists of two schemes: CacheData, which caches the data, and CacheFace, which caches the outgoing interface. We conduct extensive simulation experiments for performance evaluation and comparison with prior schemes. The simulation results show that the Pro^{NDN} can improve Interest satisfaction ratio and Interest satisfaction latency as well as reduce hop count and Content Store utilization ratio.

1. Introduction

Over the last decade, the number of devices connected to the Internet has been rapidly increasing due to the proliferation of emerging technologies such as Internet of Things, artificial intelligence, and blockchain [1]. According to Cisco Annual Internet Report [2], there will be 29.3 billion networked devices by 2023. Even though the global network performance will be significantly improved, e.g., the fixed broadband speed and mobile network connection speed will reach 110.4 Mbps and 43.9 Mbps in 2023, respectively, and the fast growth of connected devices still put high pressure on the underlying Internet infrastructure, which was developed in the 1970s. Today's Internet has exceeded all expectations for facilitating conversations between communication endpoints but shows signs of aging when it meets with next-generation content-oriented services and applications [3]. Thus, in order to keep pace with a changing world, a future Internet architecture, named data networking [4], has been regarded as the most promising Internet architecture to drive further growth and success of the future Internet.

In NDN, all communications are performed by using Interest and Data packets, both of which carry data (or content) names rather than host (or physical location) addresses. The data names are hierarchically structured like URLs; e.g., the first segment of author's paper may have the name */marshall.edu/cs/congpu/papers/ndn2020.pdf/segment1*,

where "/" delineates name components in text representations. This hierarchical structure allows applications to represent the context and relationships of data elements and facilitate traffic demultiplexing. As shown in Figure 1, to retrieve data, data consumer (e.g., PC_i) first sends out an Interest packet piggybacked with the name of desired data (e.g., X). When a router receives the Interest packet, it forwards the Interest packet toward the data producer(s) based on the information in the forwarding table. Along the forwarding path, any router (e.g., R_1, R_3, or R_5) or data producer (e.g., DS_j) who has the requested data can reply a Data packet piggybacked with the requested data. Then, the Data packet is forwarded along the reverse path of the Interest packet back to data consumer (e.g., PC_i). In addition, when the router (e.g., R_1, R_3, or R_5) receives the Data packet, it caches the piggybacked data in the caching table in order to satisfy future Interests that request the same data.

Designing and evaluating stateful forwarding and in-network caching have been a major challenge within the overall NDN research area [5]. Since NDN was proposed in 2010, there have been many research efforts focusing on this challenge and a rich literature has been developed. The literature in [6] stands out as one of the notable landmarks that sketches a basic picture of NDN's forwarding daemon and describes an initial design of stateful forwarding and in-network caching. However, the conventional stateful forwarding approach and its future variants [7, 8] fail to consider multiple network metrics when accessing the status of outgoing interface alternatives, which causes the forwarding strategy not to be adaptive and sensitive to network condition changes. In addition, the default in-network caching strategy simply relies on storing each received Data packet regardless of various caching constraints and criteria. As a result, the routers in the vicinity of data producers typically incur excessive caching overhead due to the frequent data retrieval requests from remote data consumers. Consequently, the challenge of improving stateful forwarding as well as in-network caching has attracted the attention of NDN research community.

In this paper, we propose the Pro^{NDN}, a novel stateful forwarding and in-network caching strategy for NDN networks. The Pro^{NDN} consists of multicriteria decision-making (MCDM) based Interest forwarding and cooperative data caching. The MCDM-based Interest forwarding exploits the MCDM theory to select the outgoing interface to retrieve the desired data, and thus, the forwarding strategy is adaptive and responsive to diverse network conditions. Moreover, the cooperative data caching complements the default in-network caching strategy to overcome the challenge of excessive caching overhead and efficiently support data access. Our major contribution is briefly summarized in twofold:

(1) We propose the Pro^{NDN} which comprises MCDM-based Interest forwarding and cooperative data caching. The MCDM-based Interest forwarding is to employ Technique for Order Performance by Similarity to Idea Solution (TOPSIS) to dynamically evaluate outgoing interface alternatives based on multiple network metrics and objectively select an optimal outgoing interface to forward the Interest packet. In addition, the cooperative data caching consists of two schemes: CacheData, which caches the data, and CacheFace, which caches the outgoing interface.

(2) We design the MCDM-based Interest forwarding with the consideration of extendable and flexible capability, and thus, additional network metrics can be easily included. The cooperative data caching approach is seamlessly integrated with the default in-network caching strategy, and thus, it can be regarded as additional cache policy in NDN forwarding daemon. We revisit prior forwarding and caching strategies, Con^{NDN} [6, 7] and liteNDN [8], and modify them to work in the framework for performance comparison.

We develop a customized discrete event-driven simulation framework using OMNeT++ [9] and evaluate its performance through extensive simulation experiments in terms of Interest satisfaction ratio, Interest satisfaction latency, hop count, cache hit ratio, and Content Store utilization ratio. The simulation results show that the Pro^{NDN} can improve Interest satisfaction ratio and Interest satisfaction latency as well as reduce hop count and Content Store utilization ratio, indicating a viable stateful forwarding and in-network caching strategy in NDN networks.

The rest of the paper is organized as follows. Prior forwarding and caching strategies are presented and analyzed in Section 2. In Section 3, the basic operations of NDN's stateful forwarding and in-network caching and the architecture of the Pro^{NDN} are presented. The MCDM-based Interest forwarding and cooperative data caching is presented in Sections 4 and 5, respectively. Section 6 focuses on simulation results and their analyses. Finally, concluding remarks and future research directions are provided in Section 7.

2. Related Work

In this section, we present and analyze a variety of up-to-date stateful forwarding and in-network caching strategies in NDN.

2.1. Stateful Forwarding Strategy. The authors in [8] propose a cooperative forwarding strategy for NDN networks, where routers share their information such as data names and interfaces to optimize their packet forwarding decisions and estimate the probability of each downstream path to swiftly retrieve the requested data. However, each router needs to collect the information about the names of data being exchanged from neighboring routers, which generates a large number of control messages and in turn increases the communication overhead in NDN networks. In [10], a forwarding strategy is proposed to balance the tradeoff between network overhead and performance satisfactory in Internet of Things environments. Each node overhears Data packets and learns a cost value by reinforcement and then

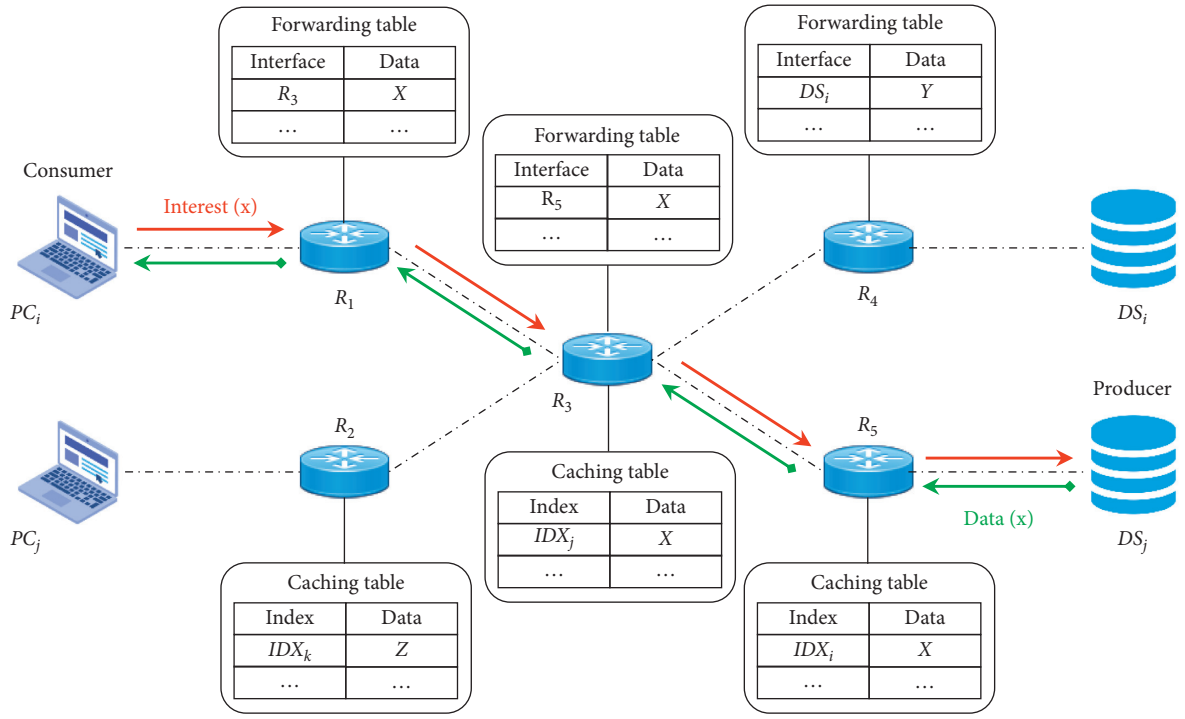


Figure 1: NDN communication model.

decides to broadcast an Interest packet with a delay according to their cost-based eligibility. The broadcast-based forwarding on the top of MAC layer can reduce communication overhead; however, a node might not be able to forward the Interest packet timely if the wireless medium is always busy. The authors in [11] propose a forwarding strategy named IFS-RL based on reinforcement learning. The IFS-RL trains a neural network model which chooses appropriate interfaces for the forwarding of Interest based on observations of the resulting performance of past decisions collected by a routing node. The IFS-RL can achieve the goal of improving throughput and packet drop rate but fails in load balancing.

In [12], a forwarding strategy is proposed for persistent Interests in NDN, where forwarding decisions are based on a combination of information from the forwarding information base and probing results. Clients issue probing Interests in order to rate paths through the network, and all probe-receiving routers can use them to evaluate the performance of already known paths, but also to explore new, possibly better paths. Nevertheless, the probing Interest packets will significantly increase network traffic and cause other issues such as traffic congestion and packet loss. The authors in [13] exploit the partially observable Markov decision process (POMDP) to design NDN request forwarding mechanism based upon the key concept of event. Since the exact optimal solution of POMDP problems in general is extremely computationally demanding, a simulation-based optimization algorithm is also proposed to find the approximate optimal solution. In [14], a deep reinforcement learning-based forwarding strategy is proposed in NDN, where the details such as data content, interface status, and network states are first collected during the forwarding process. After that, the collected information is used as input of the deep reinforcement learning for training, whose result will be used as the forwarding strategy to guide the forwarding of Interest packets. The authors in [5] provide a list of requirements of NDN forwarding plane and compare all available schemes proposed for NDN forwarding plane based on the data structure utilized. In addition, this survey paper discusses some issues, challenges, and directions in future research.

2.2. In-Network Caching Strategy. In [15], the authors propose a sum-up Bloom-filter-based request node collaboration caching (BRCC) approach for NDN networks, where different forms of caching are deployed for different types of data content. In addition, the BRCC uses the sum-up Bloom filter to enhance the data content matching rate and decrease the searching time. The simulations indicate that the BRCC can improve cache hit ratio and caching efficiency. However, the BRCC makes caching decision based on the subscriber's request frequency only, which has poor extensibility when considering additional caching criterion. Araújo et al. [16] introduce a shared caching in named data networking mobile (SCaN-Mob) strategy which aims to alleviate the side effects of producer mobility by adopting an opportunistic content caching in mobile NDNs. In the SCaN-Mob, the mobile producer carries out a round-robin selection of a device in the vicinity to store a copy of the searched content upon receiving a content request. The proposed SCaN-Mob can reach a greater content diversity, thereby increasing the likelihood of satisfying the Interest requests during the producer's unavailability periods. However, wireless devices usually have limited memory

storage, and the opportunistic caching strategy may select a device that does not have enough storage to cache the data content.

The authors in [17] present a dynamic popularity-based caching permission strategy (DPCP) for NDN networks. The DPCP takes advantage of Interest packet and Data packet to carry content popularity, so routers in the path can obtain the information about the content popularity and use dynamic popularity threshold to make cache permission policy. In addition, the DPCP deploys a cache control flag to avoid caching the same redundant copies in adjacent routers. The DPCP can reduce the amount of redundant data in the network. However, it is very challenging to set an accurate popularity threshold to make caching decision. In [18], a probability-based caching strategy with consistent harsh is proposed in NDN networks, where the caching decision is made based on the probability that is calculated by jointly considering content's popularity, node's betweenness, and distance to consumers. In [19], the authors propose a two-layer hierarchical cluster-based caching solution to improve in-network caching efficiency. A network is grouped into several clusters, and then, a cluster head is nominated for each cluster to make caching decision. However, the cluster head has to collect and allocate the information of node importance based on betweenness centrality, content popularity, and probability matrix in its cluster, which introduces a significant amount of communication overhead. Saxena et al. [20] broadly categorize caching schemes into cache placement and cache replacement. The cache placement is used to decide whether to cache the data content at the network or not, while the cache replacement is adopted to evict data from the cache when new data arrive.

2.3. Our Approach. NDN's stateful forwarding strategy decides how to effectively evaluate multiple outgoing interface alternatives and objectively choose the best interface(s) to forward the Interest packets. In summary, most prior forwarding strategies are implemented as either adaptive forwarding or context-aware forwarding, where various optimization or machine learning techniques are used to strike a balance between several performance metrics and facilitate interface adaptation. However, little attention has been paid to multicriteria decision-making (MCDM) based stateful forwarding strategy for NDN networks, where each outgoing interface alternative is evaluated in the matter of multiple network metrics and the optimal outgoing interface is chosen to forward the Interest packet based on Technique for Order Performance by Similarity to Idea Solution (TOPSIS). By taking into account multiple network metrics, the overall framework of stateful forwarding is adaptive and responsive to diverse network conditions. Another desirable feature is that the MCDM-based Interest forwarding is designed with the consideration of good extensibility and flexibility. Thus, additional network metrics can be easily included in the MCDM-based Interest forwarding for potential extension. The in-network caching is fundamentally important to support the basic concepts of NDN communication paradigm and brings several benefits, such as dissociating data from their producers, relieving the communication overhead at the data producer side, and reducing the network load and data dissemination latency. Nonetheless, little work has been done on the cooperative data caching which can overcome the challenge of excessive caching overhead and efficiently support data access in NDN networks. In addition, the cooperative data caching approach is regarded as additional cache policy in NDN forwarding daemon. Therefore, the proposed caching approach can be seamlessly integrated with the default in-network caching strategy to efficiently support data access in NDN networks.

3. Preliminaries and System Overview

In this section, we first present and analyze NDN's stateful forwarding and in-network caching, and then, we introduce the overall architecture of the proposed Pro^{NDN}.

3.1. Stateful Forwarding. As shown in Figure 2, a data consumer can retrieve data by issuing an Interest packet piggybacked with the name of desired data to the network. When a router receives the Interest packet, it first checks whether its Content Store already caches the desired data or not. Here, router's Content Store is a temporary cache of Data packets it has received. If the desired data exists in the Content Store, the router replies a Data packet piggybacked with the desired data back to the consumer along the reverse path of Interest packet. Otherwise, the router checks the name of desired data with each entry in the Pending Interest Table. In NDN, the Pending Interest Table stores the forwarded Interest packets but have not been satisfied by Data packets yet. In addition, each Pending Interest Table entry contains four components: data name, nonce, and incoming interface of the Interest packet has been received from and outgoing interface of the Interest packet has been forwarded to. If the Pending Interest Table contains an entry with the same data name and nonce as the Interest packet, the router immediately drops the Interest packet because the Interest packet that has been forwarded before is looped back. If there is an entry with matching data name and unmatching nonce, the router just adds a new entry with data name, nonce, and incoming interface without forwarding the Interest packet since this Interest packet is considered as subsequent Interest. If the data name and nonce do not match with any entry in the Pending Interest Table, the router forwards the Interest packet to an outgoing interface according to forwarding strategy and adds a new entry in the Pending Interest Table.

The forwarding strategy makes Interest packet forwarding decision based on the information stored in the Forwarding Information Base, where each entry records a name prefix and a list of outgoing interfaces together with their associated forwarding preference. The forwarding preference reflects forwarding policy as well as the cost of forwarding path which is typically calculated using certain network metrics. For example, the BestRoute [7] adopts the

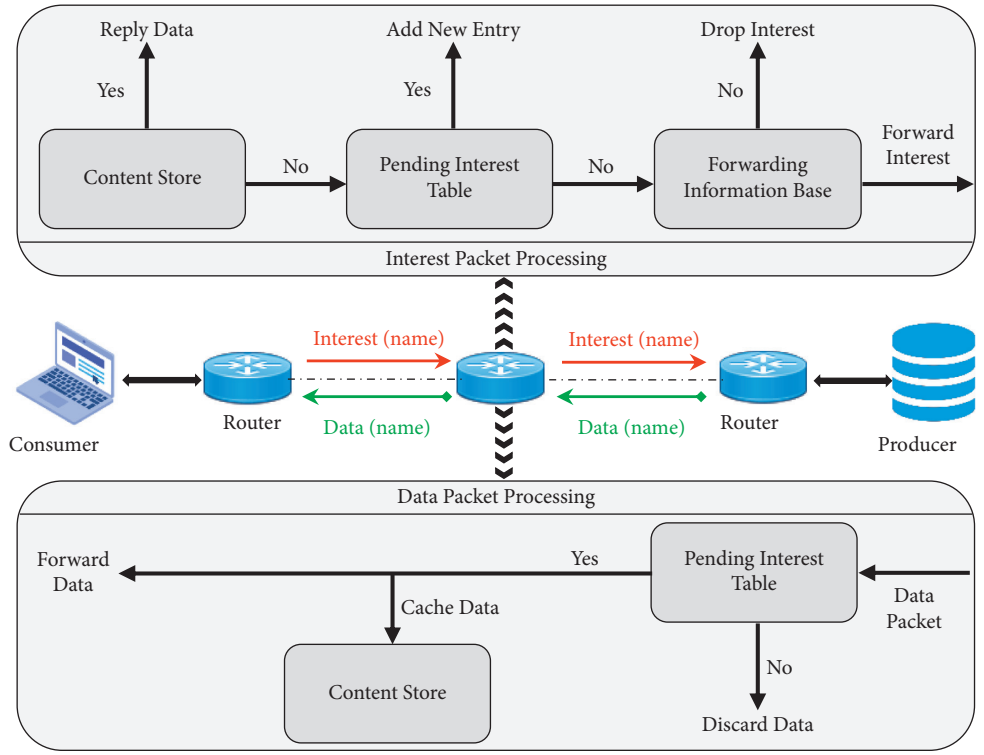


Figure 2: Interest and Data packets processing in NDN.

coloring scheme to represent the working status of each outgoing interface, based on which the forwarding strategy selects the best outgoing interface to forward an Interest packet. For each name prefix, all outgoing interfaces are ranked based on Interest rate limit, and the highest ranked Green outgoing interface is always selected to forward an Interest packet. If there is no Green outgoing interface, the highest ranked Yellow outgoing interface is adopted. The Red outgoing interfaces are never used because they cannot bring data back. The forwarding strategy in the BestRoute can improve the link utilization. However, the BestRoute fails to detect and respond to network condition changes timely because it only considers one network metrics (i.e., Interest rate limit) to make forwarding decision. For instance, if the Interest packet forwarding rate reaches the rate limit, the outgoing interface will experience traffic congestion sooner or later, which leads to the fact that the second-ranked Green outgoing interface would be the best option for Interest packet forwarding. Thus, the highest ranked Green outgoing interface may not always be the best option with various changes in-network conditions. In summary, the forwarding strategy is playing an important role in NDN forwarding plane. In order to improve the network performance and respond to network condition changes accurately and astutely, the forwarding strategy should take into account multiple network metrics to make Interest packet forwarding decision.

3.2. In-Network Caching. When the data producer or the router who caches the desired data in the Content Store receives the Interest packet, it replies a Data packet piggybacked with the desired data back to the data consumer. When a router receives the Data packet from an upstream router or the data producer, it first searches the piggybacked data name in the Pending Interest Table. If an entry with matching data name is found in the Pending Interest Table, the router forwards the Data packet to all stored incoming interfaces, caches a copy of piggybacked data in the Content Store, and removes all entries with matching data name from the Pending Interest Table. Otherwise, the router drops the Data packet because the data are unsolicited and may pose security risks to the forwarder. However, these are also cases when unsolicited Data packets need to be stored in the Content Store. In order to purge the stale entry in the Pending Interest Table, an entry lifetime is assigned to each entry. When the lifetime expires, the entry is removed from the Pending Interest Table.

The default in-network caching relies on storing each received Data packet disregarding various caching constraints and criteria. For large-scale network with a significant amount of data retrieval traffic, the routers that are located in the vicinity of data producers will receive an excessive number of Interest packets from remote data consumers, which definitely causes enormous caching overhead. As a result, the cache space of these routers can be exhausted in vain. To tackle the issue of excessive caching overhead for the routers in the vicinity of data producers, the liteNDN [8] implements a decision-making mechanism to proactively decide upon caching the received Data packets, where the routers located close to a certain data producer can completely avoid caching Data packets from this data producer. The liteNDN can reduce the caching overhead.

However, it does not consider the popularity of data in the cache decision-making process and completely discards the default in-network caching strategy. In one word, the caching is a common technique to improve the performance of data access. Thus, it should be treated with respect to efficiently support data access in NDN networks.

3.3. Pro^{NDN} Architecture. As shown in Figure 3, the Pro^{NDN} comprises two main components, namely, MCDM-based Interest forwarding and cooperative data caching. In the MCDM-based Interest forwarding strategy, when a router receives an Interest packet to forward, it evaluates all outgoing interface alternatives based on the combination of multiple network metrics and selects an optimal outgoing interface to forward the Interest packet. To be specific, the router first establishes a decision matrix with the up-to-date network metrics information and calculates the weighted normalized decision matrix by multiplying the normalized decision matrix by the relative weights of multiple network metrics. And then, the router calculates the forwarding index of each outgoing interface alternative and chooses the outgoing interface with the highest ranked forwarding index to forward the Interest packet. In the cooperative data caching strategy, when a router receives a Data packet, it will decide whether to apply CacheData, CacheFace, or default in-network caching based on the predefined rules. In short, if the piggybacked data are popular, the router caches the received Data packet by adopting CacheData. Otherwise, the router chooses to apply CacheFace by caching the outgoing interface toward the border router who has the data if its distance to the border router is shorter than its distance to the data producer. If both CacheData and CacheFace are not applicable, the router adopts the default in-network caching. More details about the proposed MCDM-based Interest forwarding and cooperative data caching strategies are presented as follows. Table 1 lists all notations used in this paper.

4. MCDM-Based Interest Forwarding

The basic idea of the MCDM-based Interest forwarding is to employ Technique for Order Performance by Similarity to Idea Solution (TOPSIS) to dynamically evaluate outgoing interface alternatives based on multiple network metrics and objectively select an optimal outgoing interface to forward the Interest packet. The TOPSIS is a multicriteria decision-making model to identify the best alternative that is nearest to the positive-ideal solution and farthest from the negative-ideal solution [21]. When a router receives an Interest packet to forward, it evaluates all outgoing interface alternatives based on the up-to-date network metrics information and calculates the forwarding index of each outgoing interface. Based on the forwarding index, the router ranks all outgoing interface alternatives and selects the highest ranked outgoing interface to forward the Interest packet. The detailed design of the MCDM-based Interest forwarding is provided as follows.

First, the router establishes a decision matrix with the up-to-date network metrics information for the ranking of outgoing interface alternatives. The structure of the decision matrix can be expressed as follows:

$$M = \begin{array}{c} \\ \mathrm{AI}_1 \\ \mathrm{AI}_2 \\ \vdots \\ \mathrm{AI}_i \\ \vdots \\ \mathrm{AI}_n \end{array} \begin{array}{c} \begin{array}{cccccc} \mathrm{PM}_1 & \mathrm{PM}_2 & \cdots & \mathrm{PM}_j & \cdots & \mathrm{PM}_n \end{array} \\ \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1j} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2j} & \cdots & x_{2n} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ x_{i1} & x_{i2} & \cdots & x_{ij} & \cdots & x_{in} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mj} & \cdots & x_{mn} \end{bmatrix} \end{array}, \quad (1)$$

where AI_i represents the i^{th} outgoing interface alternative, $i = 1, 2, \ldots, m$; PM_j denotes the j^{th} network metrics, $j = 1, 2, \ldots, n$; and x_{ij} is a crisp value of the j^{th} network metrics related to the i^{th} outgoing interface alternative. Second, the router generates the normalized decision matrix $M^{\mathrm{norm}}(=x_{ij}^{*})$ according to

$$x_{ij}^{*} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^{n} x_{ij}^{2}}}, \quad i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n. \quad (2)$$

Since the scales of measurement for multiple network metrics are not unique, it is important to normalize the decision matrix to make crisp values comparable to each other. Third, the router calculates the weighted normalized decision matrix by multiplying the normalized decision matrix by the relative weights of multiple network metrics. The weighted normalized decision matrix $M^{\mathrm{wgt}}(=x_{ij}^{\oplus})$ is calculated as

$$x_{ij}^{\oplus} = w_j \times x_{ij}^{*}, \quad i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n, \quad (3)$$

where w_j represents the relative weight of the j^{th} network metrics. The rationale behind the design of w_j is to adjust the effect of the j^{th} network metrics for subjective preference. The relative weights of multiple network metrics can be determined by applying analytic network process (ANP) [21]. Fourth, the router calculates the separation measurement using m-dimensional Euclidean distance. The separation between the i^{th} outgoing interface alternative and positive-ideal solution, denoted as Sol_i^{+}, is given as

$$\mathrm{Sol}_i^{+} = \sqrt{\sum_{j=1}^{n} \left(x_{ij}^{\oplus} - \max\left(\mathrm{PM}_j\right)\right)^2}, \quad i = 1, 2, \ldots, m. \quad (4)$$

Similarly, the separation between the i^{th} outgoing interface alternative and negative-ideal solution, denoted as Sol_i^{-}, is as follows:

$$\mathrm{Sol}_i^{-} = \sqrt{\sum_{i=1}^{n} \left(x_{ij}^{\oplus} - \min\left(\mathrm{PM}_j\right)\right)^2}, \quad i = 1, 2, \ldots, m. \quad (5)$$

Based on the separation measurements, the router can calculate the relative closeness of the i^{th} outgoing interface alternative to the idea solution as follows:

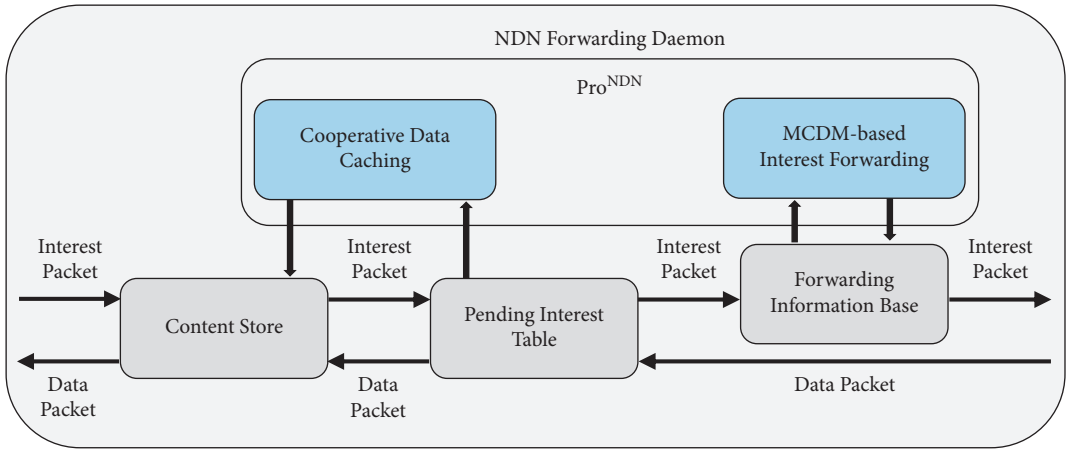


Figure 3: Architecture of Pro^{NDN}.

Table 1: Notations.

Notation	Meaning
M	Decision matrix
AI_i	The i^{th} outgoing interface alternative
PM_j	The j^{th} network metrics
x_{ij}	Crisp value of PM_j related to AI_i
M^{norm}	Normalized decision matrix
M^{wgt}	Weighted normalized decision matrix
w_j	Relative weight of PM_j
Sol_i^+	Separation between AI_i and positive-ideal solution
Sol_i^-	Separation between AI_i and negative-ideal solution
$\mathbb{I}_i$	Forwarding index of AI_i
$\triangle^{In\longrightarrow}$	System parameter for adopting CacheData
$\triangle^{hop}$	System parameter for adopting CacheFace
H (producer)	The number of hops to the data producer
H (border)	The number of hops to the boarder router

$$\mathbb{I}_i = \frac{Sol_i^-}{Sol_i^- + Sol_i^+}, \quad i = 1, 2, \ldots, m, \tag{6}$$

where the $\mathbb{I}_i$ is considered as the forwarding index of the i^{th} outgoing interface alternative. The $\mathbb{I}_i$ lies between 0 and 1, and the larger the forwarding index value means the better the overall performance of the i^{th} outgoing interface alternative. Finally, the router ranks the forwarding indexes of all outgoing interface alternatives, and the highest ranked outgoing interface will be the optimal one to forward the Interest packet.

For example, suppose that a route has four outgoing interface alternatives (AI_{1-4}) to choose and forward the Interest packet. We consider interface utilization ratio, round-trip time (RTT), and NACK ratio as real-time network metrics to calculate the forwarding index of each outgoing interface alternative. The decision matrix containing the crisp value of network metrics is shown in Table 2. According to equations (2) and (3), the normalized decision matrix and the weighted normalized decision matrix is calculated and presented in Tables 3 and 4, respectively. Here, the relative weight of interface utilization ratio, RTT, and NACK ratio is set to 0.3, 0.4, and 0.3, respectively. After that, the separation measurement between each outgoing interface alternative and the positive and negative-ideal solutions can be calculated by using the data in Table 4, and related results are shown in Table 5.

Table 2: Decision matrix for interface alternatives.

Interface ID	Interface util. ratio (%)	RTT (ms)	NACK ratio (%)
AI_1	75	73	25
AI_2	25	45	10
AI_3	62	67	19
AI_4	84	80	30

Table 3: Normalized decision matrix for interface alternatives.

Interface ID	Interface util. ratio	RTT	NACK ratio
AI_1	0.69699	0.67840	0.23233
AI_2	0.47673	0.85812	0.19069
AI_3	0.66494	0.71856	0.20377
AI_4	0.70107	0.66769	0.25038

Table 4: Weighted normalized decision matrix for interface alternatives.

Interface ID	Interface util. ratio	RTT	NACK ratio
AI_1	0.20910	0.27136	0.06970
AI_2	0.14302	0.34325	0.05721
AI_3	0.19948	0.28742	0.06113
AI_4	0.21032	0.26708	0.07511

In the final ranking stage, by using equation (6), the forwarding index of each outgoing interface alternative is calculated. The calculated forwarding indexes are ranked and listed in Table 6. According to the forwarding index, the ranking order of four outgoing interface alternatives is AI_2, AI_3, AI_1, and AI_4, which indicates that AI_2 is the best outgoing interface candidate to choose and forward the Interest packet. Here, AI_2 has the lowest interface utilization ratio, 25%, shortest RTT, 45 ms, and smallest NACK ratio, 10%. Major operations of the MCDM-based Interest forwarding are summarized in Algorithm 1.

Table 5: Separation distances for interface alternatives.

Interface ID	Sol^+	Sol^-
AI_1	0.07210	0.06739
AI_2	0.06964	0.07617
AI_3	0.05857	0.06014
AI_4	0.07617	0.06964

Table 6: Forwarding index ranking for interface alternatives.

Rank	Interface ID	Forwarding index
1	AI_1	0.52239
2	AI_2	0.50661
3	AI_3	0.48312
4	AI_4	0.47761

5. Cooperative Data Caching

In NDN, since each Data packet only carries a data name, it is said to be independent of who requested or from where it is retrieved [4]. In order to quickly satisfy future Interest packet requesting the same data, a router can choose to cache a copy of received Data packet in its Content Store when it receives a Data packet. NDN's default in-network caching [6] is designed to store each received Data packet regardless of various caching constraints and criteria. The default caching strategy is incredibly simple and easy to implement. However, the potential issue is that the routers located in the vicinity of data producers will receive an excessive number of Interest packets from remote data consumers, which definitely causes enormous caching overhead. For large-scale network with a significant amount of data retrieval traffic, the Content Store of these routers can be exhausted in vain, which in turn causes more frequent cache replacement operations. Moreover, the routers in the vicinity of data producers may have a worst-case caching overhead of $\Theta(n)$, where n is the total number of Interest packets requesting different data from all data consumers. In light of these, we propose a cooperative data caching strategy to overcome the challenge of excessive caching overhead and efficiently support data access in NDN networks. The cooperative data caching strategy consists of two schemes, CacheData and CacheFace, to complement the default in-network caching strategy. In the following, we present CacheData and CacheFace with more details.

In the CacheData, the router caches the received Data packet if more than $\triangle^{\overrightarrow{In}}$ number of different incoming interfaces request the piggybacked data. Here, $\triangle^{\overrightarrow{In}}$ is a system parameter, $\triangle^{\overrightarrow{In}} = 1, 2, \ldots, n$. The rationale behind the design of CacheData is that if the data are popular, i.e., many Interest packets from different incoming interfaces request the data, the router should cache the received Data packet. The basic idea of CacheData can be explained by using Figure 4. Suppose that the data consumer PC_1 and PC_2 are interested in data d_i and send out Interest packets to retrieve d_i through the router R_3. Here, we assume that $\triangle^{\overrightarrow{In}} = 1$. R_3 receives Interest packets from two different incoming interfaces connected with PC_1 and PC_2, respectively, and thus, it should cache a copy of d_i when receiving the Data packet according to the CacheData. Since all Interest packets received by the router R_1 come from R_2, which in turn come from R_3, and thus, R_1 and R_2 do not cache the received Data packet. As another example, considering that the data consumer PC_1 and PC_4 send out Interest packets to retrieve the data d_i. With the CacheData scheme, the router R_2 should cache the Data packet, whereas R_1, R_3, and R_4 need not to do so. In summary, the CacheData are designed to cache the Data packet conservatively. In some rare situations, for instance, when most of data consumers are interested in certain data at the same time, the CacheData might decrease the cache hit ratio because the data are not cached at every intermediate router. However, we do not assume that certain data are interested by all data consumers concurrently in NDN networks. Thus, the CacheData can be adopted to solve the challenge of excessive caching overhead as well as efficiently support data access in NDN networks.

In the CacheFace, the router caches the outgoing interface toward the border router who has the data and uses it to redirect future Interest packets if its distance (i.e., hop count) to the border router is $\triangle^{\text{hop}}$ hop(s) shorter than its distance to the data producer. Here, $\triangle^{\text{hop}}$ is the number of hops that a cached interface can save and is denoted as

$$\triangle^{\text{hop}} = H(\text{producer}) - H(\text{border}), \tag{7}$$

where $H(\text{producer})$ is the number of hops to the data producer and $H(\text{border})$ is the number of hops to the border router. In this paper, we define the border router as the router that is directly connected with data consumer(s). In Figure 4, R_3 and R_4 are considered as border routers. The rationale behind the design of CacheFace is that the data retrieval latency can be reduced if the data can be obtained through a shorter distance. For example, in Figure 4, suppose that the data consumers PC_1 and PC_2 have requested the data d_i through the border router R_3. Here, we assume that $\triangle^{\text{hop}} = 1$. When the router R_2 receives and forwards the Data packet piggybacked with d_i to R_3, R_2 knows that R_3 has a copy of d_i. Later, if the data consumer PC_4 requests d_i through the router R_4, which in turn through R_2, R_2 knows that the data producer is two hops away, whereas the border router R_3 who has d_i is only one hop away. Therefore, R_2 forwards the Interest packet to R_3 instead of R_1 to retrieve d_i. To properly make CacheFace decision, it is necessary to embed the hop count information in the header of each NDN packets. Thanks to the adoption of type-length-value (TLV) format, which is an encoding scheme used for optional information element in NDN, the hop count can be easily added as a new field and type in the NDN packets [22]. When the data consumer issues an Interest packet, it initializes the hop count value to zero. When a router receives the Interest packet, it increases the hop count by one and forwards the Interest packet. Thus, each router along the forwarding path knows its hop count distance to the data consumer and boarder router when receiving the Interest packet. The same idea will be applied to the Data packet. In the CacheFace, the router only needs to cache the outgoing

Result: The highest ranked outgoing interface
M, M^{norm}, x_{ij}, x^{*}_{ij}, $m^{\text{wgt}}w_j$, $x^{\oplus}_{ij}$, Sol^{+}_{i}, Sol^{-}_{i}, $\mathbb{I}_i$:
Defined before;
R_k and $In^{\leftarrow}_{j}$: A router R_k and an outgoing interface j;
$\text{RTT}_k[j]$: A round-trip time of $\text{In}^{\leftarrow}_{j}$ at R_k;
$\text{NACK}_k[j]$: A NACK ratio of $In^{\leftarrow}_{j}$ at R_k;
$\text{IUR}_k[j]$: An interface utilization ratio of $In^{\leftarrow}_{j}$ at R_k;
$pkt[name, type, hop]$: A packet containing a name of data content (*name*), packet type (*type*), and a hop count to packet initiator (*hop*). Here, *type* can be either *Interest*, *Data*, or *NACK*;
When R_k receives a $pkt[name, Da\ ta, hop]$ from $In^{\leftarrow}_{j}$:
Update $RTT_k[j]$;
When R_k receives a $pk[[name, NACK, hop]$ from $In^{\leftarrow}_{j}$:
Update $NACK_k[j]$;
When R_k forwards a $pkt[name, Interest, hop]$ through $In^{\leftarrow}_{j}$:
Update $IUR_k[j]$;
When R_k has an Interest packet to forward:
Retrieve RTT_k, $NACK_k$, and IUR_k; Create M;
Calculate M^{norm} according to equation (2);
Calculate M^{wgt} according to equation (3);
Calculate Sol^{+} and Sol^{-} according to equations (4) and (5), respectively;
Calculate $\mathbb{I}$ according to equation (6);
Rank all outgoing interface alternatives in terms of $\mathbb{I}$;
Select the highest ranked outgoing interface to forward Interest;

Algorithm 1: The proposed MCDM-based Interest forwarding.

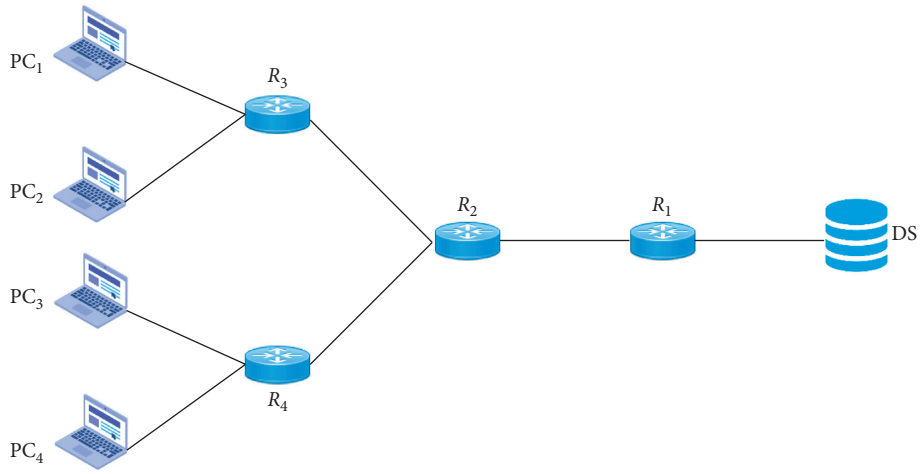


Figure 4: Small-scale tree topology.

interface toward the border router when it is closer to the border router than to the data producer. For instance, when the Data packet piggybacked with d_i is forwarded back to both PC_1 and PC_2 along the path $R_1 \longrightarrow R_2 \longrightarrow R_3$, R_1 does not need to cache the outgoing interface toward the border router R_3 because R_1 is closer to the data producer than to R_3.

In the cooperative data caching strategy, when a router receives a Data packet, it decides whether to apply CacheData, CacheFace, or default in-network caching based on the following rules:

First, the CacheData is adopted if the router receives the Interest packets from more than $\triangle^{In^{\longrightarrow}}$ number of different incoming interfaces.

Second, if the CacheData is not applicable, the CacheFace is applied if the router's distance to the border router is $\triangle^{\text{hop}}$ hop(s) shorter than its distance to the data producer.

Third, if both CacheData and CacheFace are not applicable, the default in-network caching is adopted.

Major operations of the cooperative data caching are summarized in Algorithm 2.

6. Performance Evaluation

6.1. Simulation Testbed and Benchmarks. We conduct extensive simulation experiments using OMNeT++ [9] to evaluate the performance of the Pro^{NDN}. 100 nodes are

$\triangle^{In^{\longrightarrow}}$, $\triangle^{hop}$, $pkt[name, type, hop]$, R_i, PC_i, and d_i:
Defined before;
PIT_j: The Pending Interest Table at R_j;
$PIT_j[In^{\longrightarrow}, d_i]$: The set of incoming interfaces associated with data d_i in the Pending Interest Table at R_j. Here, $In^{\longrightarrow}$ is the set of incoming interfaces that the Interest packets have been received from;
CS_j: The Content Store at R_j;
FIB_j: The Forwarding Information Base at R_j;
$H_j(producer)$: The number of hops between the data producer and R_j;
$H_j(border)$: the number of hops between the boarder router and R_j;
When R_j receives a Data packet $pkt[d_i, Data, hop]$:
if $d_i \in PIT_j$ **then**
  Forward $pkt[d_i, Data, hop]$ to $PIT_j[In^{\longrightarrow}, d_i]$
  if *count* $(PIT_j[In^{\longrightarrow}, d_i]) > \triangle^{In^{\longrightarrow}}$ **then**
    Cache d_i in CS_j
    /* Apply CacheData */
  else
    if $(H_j(producer) - H_j(border)) > \triangle^{hop}$ **then**
      Cache $PIT_j[d_i].In^{\longrightarrow}$ in FIB_j
      /* Apply CacheFace */
    else
      Cache d_i in CS_j
      /* Apply default in-network caching */
    end
  end
  Remove $PIT_j[In^{\longrightarrow}, d_i]$ from PIT_j
else
  Drop $pkt[d_i, Data, hop]$
  /* Unsolicited Data packet */
end

Algorithm 2: The proposed cooperative data caching.

randomly distributed in the network area, where 5 nodes are selected to serve as data consumers and data producers, respectively. To generate random network topologies, we employ the network topology generator BRITE [23], which is a parametrized topology generator that can be used to study the relevance of possible causes for power laws and other metrics observed in Internet topologies. Table 7 specifies the configured network connectivities: low connectivity, medium connectivity, and high connectivity, where the second column identifies an integer number of links per router. Figure 5 illustrates three sample random network topologies with different network connectivities. In addition, the Interest packet rate of each data consumer and the size of packet is set to 5 pkt/sec and 512 Bytes, respectively. In the existing literature, various packet sizes (i.e., 256, 512, and 1024 Bytes) have been adopted to evaluate the effect of packet size in NDN [24]. Thus, a medium packet size of 512 Bytes is adopted as a representative in this paper. In [25,26], the Interest packet rate and the size of packet is set to 200 pkt/sec and 1040 Bytes, respectively. So the total traffic rate is 208,000 Bytes per second. In our simulation, the total traffic rate is 2,560 Bytes per second. Thus, we believe that the value of customized system parameters such as Interest packet rate and the size of packet are within a reasonable range. The total simulation time is set to 500 seconds, and $\triangle^{In^{\longrightarrow}} = 1$ and $\triangle^{\text{hop}} = 1$ are adopted. In order to obtain steady performance results, each simulation scenario is repeated 10 times with different randomly generated seeds. In this paper, we measure the performance in terms of Interest satisfaction ratio, Interest satisfaction latency, hop count, cache hit ratio, and Content Store utilization ratio by changing key simulation parameters, including network connectivity and number of link failures.

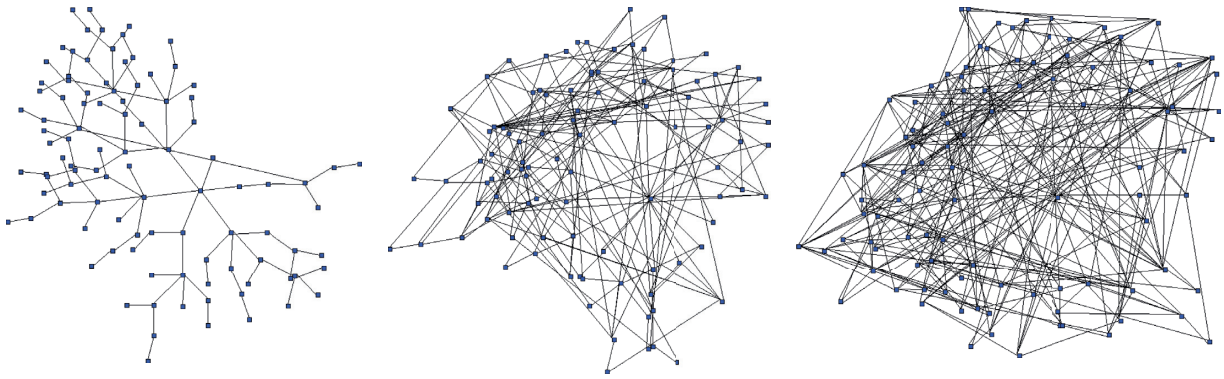
Interest satisfaction ratio: the Interest satisfaction ratio is defined as the ratio between the total number of retrieved Data packets and the total number of issued Interest packets.

Interest satisfaction latency: the Interest satisfaction latency is the averaged elapsed time when the data consumers issue the Interest packets to when the data consumers receive the Data packets.

Hop count: the hop count is calculated as the total number of links Data packets traversed to satisfy issued Interest packets divided by the total number of Data packets.

Cache hit ratio: the cache hit ratio is the ratio of the total number of satisfied Interest packets by Content Store to the total number of received Interest packets.

Content Store utilization ratio: the Content Store utilization ratio is calculated as the total number of cached Data packets divided by the size of Content Store.



(a) (b) (c)

Figure 5: Randomly generated sample network topologies with different network connectivities defined in Table 7. (a) Low connectivity. (b) Medium connectivity. (c) High connectivity.

Table 7: Network connectivity.

Connectivity	Number of links per router $\wp$
Low connectivity	1
Medium connectivity	2
High connectivity	3

$\wp$, the number of neighbor routers to which a new router connects when it joins the network [23].

We revisit prior forwarding and caching strategies, Con^{NDN} [6, 7] and liteNDN [8], and modify them to work in the framework for performance comparison. The basic idea of these two benchmark schemes is briefly discussed as follows:

Con^{NDN}: the Con^{NDN} classifies outgoing interfaces based on a coloring scheme. Outgoing interfaces are classified as Green, Yellow, and Red, which indicates that the outgoing interfaces can bring data, may or may not bring data, and cannot bring data, respectively. Interest packets are forwarded to the highest-ranked Green outgoing interface. If no Green outgoing interface is available, the highest-ranked Yellow outgoing interface is chosen to forward Interest packets. In addition, the caching strategy of Con^{NDN} relies on storing each received Data packet.

liteNDN: the liteNDN comprises two main components, including cooperative forwarding and heuristic-based caching. The former component leverages shared data names and outgoing interfaces among routers to estimate the most probable paths toward cached versions of the requested data. The latter component implements a decision-making mechanism to proactively decide upon caching the received Data packets, where the routers located close to a certain data producer can completely avoid caching Data packets from this data producer.

6.2. Simulation Results and Analysis. First, we measure the performance of Interest satisfaction ratio against network connectivity, number of link failures, and simulation time in Figure 6. As shown in Figure 6(a), the Interest satisfaction ratio of Con^{NDN}, liteNDN, and Pro^{NDN} increases as the network connectivity increases, where the number of link failures is set to zero. With a higher network connectivity, the number of neighbor routers to which a new router connects increases, and thus, the network becomes more denser. In the denser network, each router has more outgoing interface alternatives to select and forward the Interest packets, and the traffic of Interest packets is distributed among multiple paths without causing traffic congestion. As a result, data producers receive more Interest packets and then reply more Data packets, which causes the Interest satisfaction ratio to increase. The Pro^{NDN} outperforms Con^{NDN} and liteNDN. Since the Pro^{NDN} evaluates outgoing interface alternatives based on multiple network metrics and objectively select an optimal outgoing interface to forward the Interest packets, more Interest packets can be delivered to data producers. Correspondingly, more Data packets will be replied back to data consumers along the reverse path of Interest packets; thus, a higher Interest satisfaction ratio is observed. The liteNDN shows a higher Interest satisfaction ratio than that of Con^{NDN}. This is because each router reroutes the Interest packets to closer router who has Data packets based on the shared data names and interfaces information, more Data packets can be received through shorter routes, and a higher Interest satisfaction ratio can be achieved than Con^{NDN}. In Figure 6(b), the Interest satisfaction ratio of all three schemes decrease when the number of link failures increases. This is because the Interest or Data packets will get lost during the transmission when the link failure happens, and a less number of Interest or Data packets can be received. As a result, a lower Interest satisfaction ratio is obtained. However, the Pro^{NDN} still provides

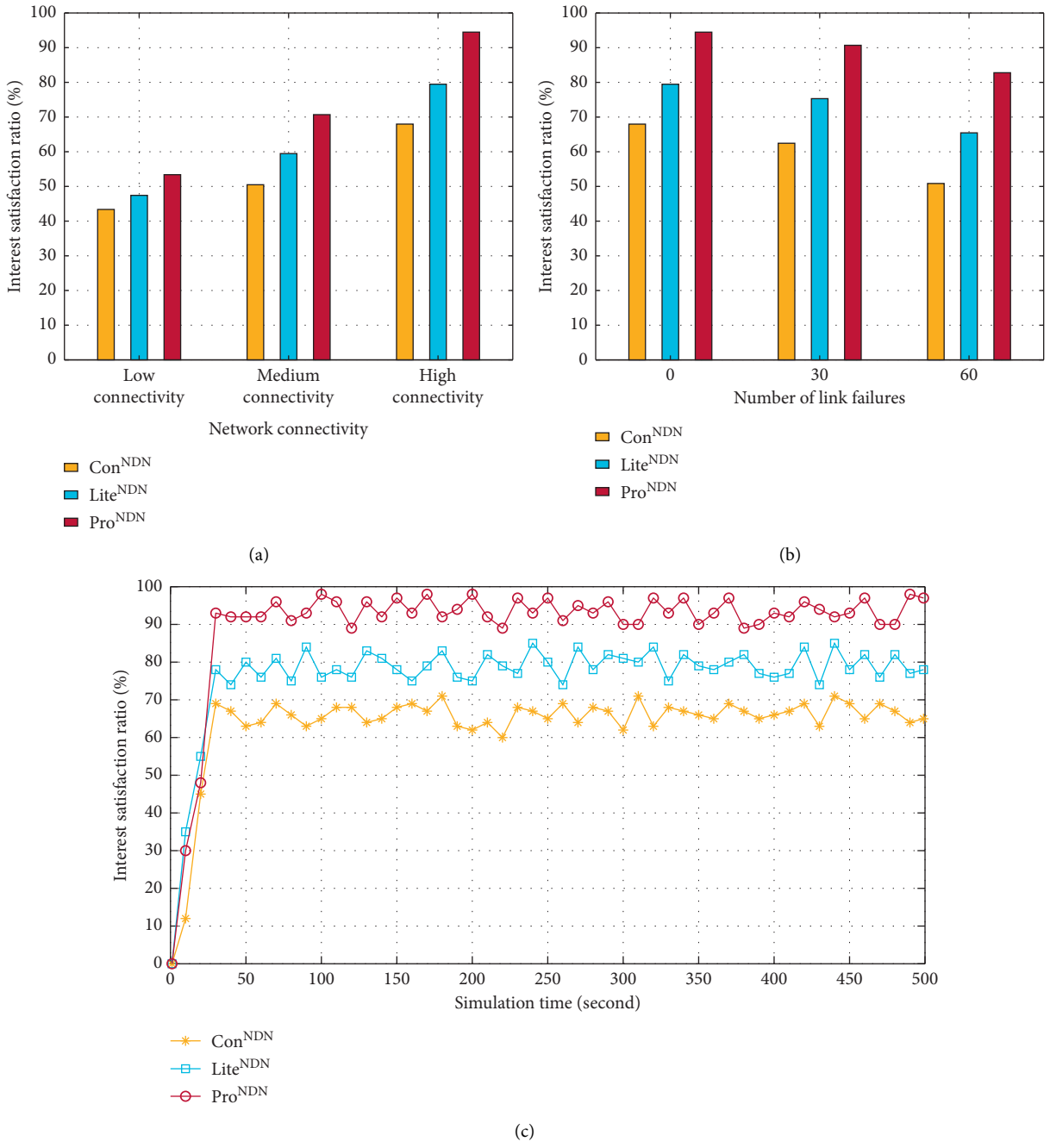


Figure 6: The performance of Interest satisfaction ratio against network connectivity, number of link failures, and simulation time.

the highest Interest satisfaction ratio because it can detect the changes in network conditions quickly and select more reliable outgoing interfaces to forward Interest packets. Thus, more Interest packets can be received by data producers; in turn, more Data packets will be replied back to data consumers and a higher Interest satisfaction ratio can be achieved. Figure 6(c) shows the changes in Interest satisfaction ratio as the simulation time elapses.

Second, we measure the performance of hop count by varying network connectivity and the number of link failures in Figure 7. In Figure 7(a), the hop count slightly decreases as the network connectivity increases. As each router has more neighbors, it more likely finds a shorter path to forward the Interest packets to data producers. Since the Data packets are replied along the reverse path of Interest packets, the number of links Data packets traversed to satisfy issued Interest packets decrease and a decreasing hop count is observed. The Pro^{NDN} obtains the lowest hop count because it evaluates the performance of RTT to select the outgoing interface, and a shorter path with a lower RTT can be identified to forward Interest packets. Thus, the lowest hop count can be achieved by Pro^{NDN} compared to Con^{NDN} and liteNDN. In addition, since each router might apply CacheFace and reroute the Interest packets to closer boarder

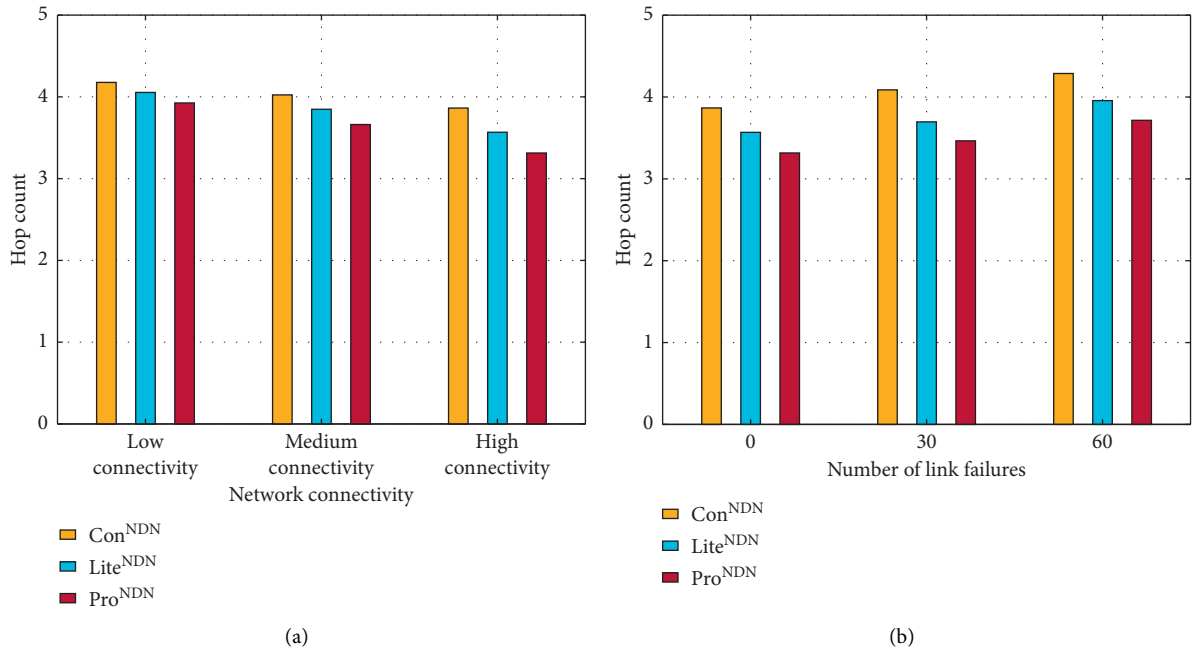


Figure 7: The performance of hop count against network connectivity and number of link failures.

router who has the Data packet, the Interests packets can be satisfied through a shorter route, which causes the hop count to decrease as well. As shown in Figure 7(b), the hop count of Pro^{NDN}, Con^{NDN}, and liteNDN increases linearly when the number of link failures increases. Since there are more link failures in the network, Interest packets might be forwarded through a longer path, which results in an increasing hop count. The Con^{NDN} shows the highest hop count because it only considers the link limit rate to select outgoing interface. The liteNDN achieves a smaller hop count than Con^{NDN}. This is because the liteNDN replies on sharing data names and interfaces knowledge among neighbor routers to select a shorter route to forward Interest packets.

Third, we measure the performance of Interest satisfaction latency by varying network connectivity and the number of link failures in Figure 8. As shown in Figure 8(a), the overall Interest satisfaction latency decreases as the network connectivity increases. When the network becomes more connected, data consumers can easily find a shorter route to forward Interest packets and retrieve the desired data from data producers. Since the Data packets will also traverse back to data consumers along the shorter route, a shorter Interest satisfaction latency can be achieved. Most importantly, the Pro^{NDN} still outperforms Con^{NDN} and liteNDN because it considers multiple network metrics to select the best outgoing interface. In addition, the Pro^{NDN} adopts CacheData and CacheFace to help retrieve the desired data through shorter routes, where the Interest packets can be satisfied by intermediate router or rerouted to closer boarder router who has the data. In Figure 8(b), the Interest satisfaction latency increases as the number of link failures increases. When there are more link failures in the network, the Interest packets might need to be forwarded through a reliable but longer routes. Thus, a higher Interest satisfaction latency is obtained. However, the Pro^{NDN} still shows a lower Interest satisfaction latency compared to that of Con^{NDN} and liteNDN.

Fourth, we measure the performance of cache hit ratio and Content Store utilization ratio by varying network connectivity in Figure 9. It is shown in Figure 9(a) that the cache hit ratio of Con^{NDN} is much higher than that of Pro^{NDN} and liteNDN. This is because the Con^{NDN} adopts the conventional caching strategy in the network, where each router stores every received Data packet. As a result, the Data packets are cached at every intermediate router, which can improve the cache hit ratio beyond all doubts. The Pro^{NDN} shows a higher cache hit ratio than liteNDN. Since the Pro^{NDN} can satisfy the Interest packets through either CacheData or CacheFace, a higher cache hit ratio can be obtained. In liteNDN, the router can reroute the Interest packets to the routers who have the desired data only if it has the information of data names and interfaces. Otherwise, the router has to forward the Interest packets to data producers to retrieve the desired data. Thus, the liteNDN delivers the lowest cache hit ratio. In Figure 9(b), it is not surprising to see that the routers in the vicinity of data producers have 100% Content Store utilization ratio in Con^{NDN}. The reason is that the Con^{NDN} adopts default in-network caching strategy which relies on storing each received Data packet disregarding various caching constraints and criteria. Since both Pro^{NDN} and liteNDN adopts caching strategies to prevent the routers in the vicinity of data producers from caching every received Data packet, a much lower Content Store utilization ratio is achieved compared to Con^{NDN}. The Pro^{NDN} shows a higher Content Store utilization ratio than that of liteNDN because the Pro^{NDN} caches the popular Data packets.

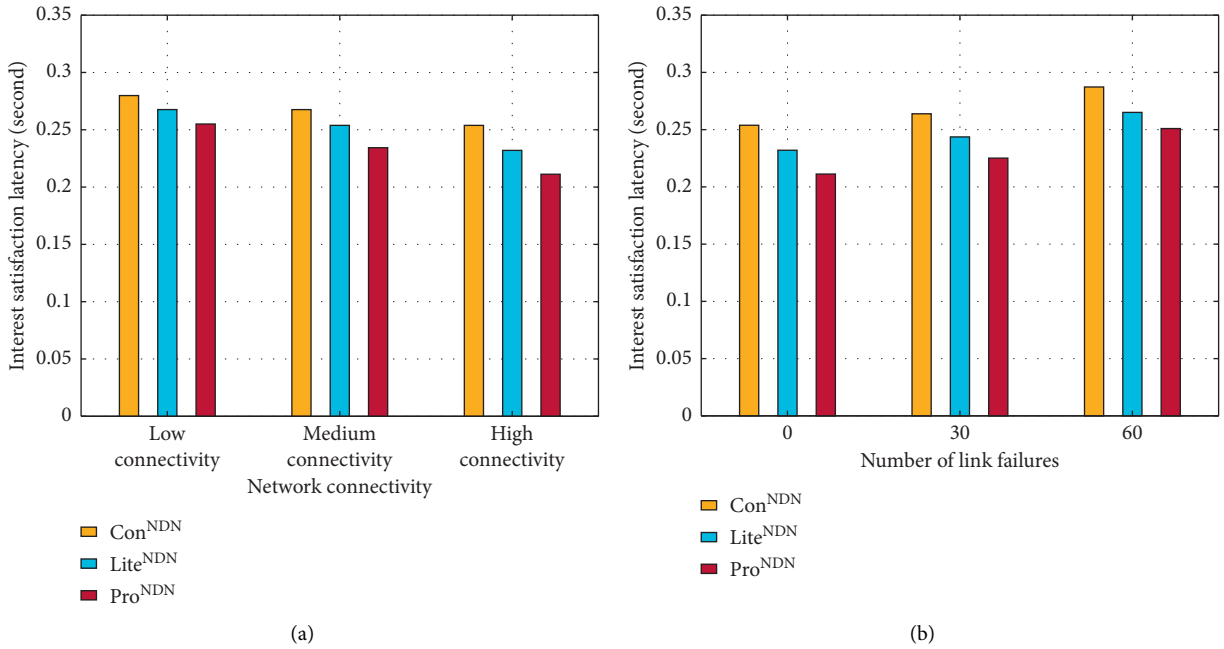


FIGURE 8: The performance of Interest satisfaction latency against network connectivity and number of link failures.

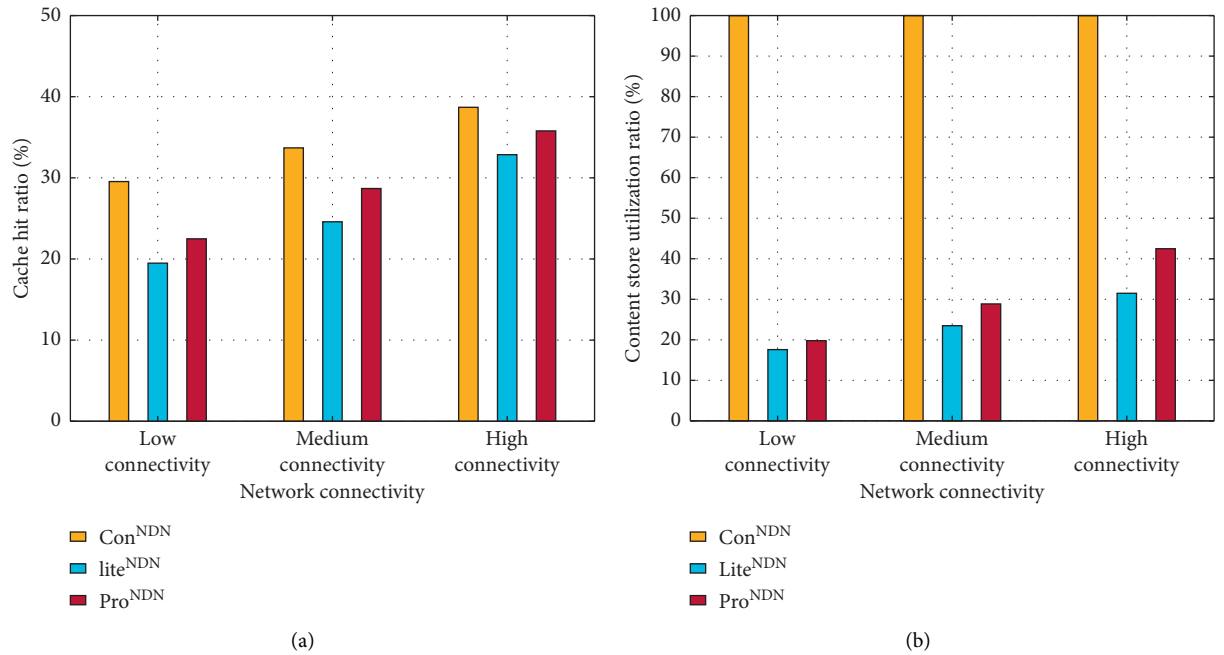


FIGURE 9: The performance of cache hit ratio and Content Store utilization ratio against network connectivity.

Fifth, we measure the performance of Interest satisfaction ratio by changing the number of nodes in Figure 10. Overall, as the number of nodes in the network increases from 100 to 200, the Interest satisfaction ratio of all three schemes increase. The rationale is that more neighbor nodes can be selected to forward the Interest packets in the network. As a result, data producers can receive more Interest packets and then reply more Data packets, which causes the Interest satisfaction ratio to increase. When the number of nodes is increased from 160 to 200, a slight increase of Interest satisfaction ratio is observed. However, the Pro^{NDN} still outperforms other two schemes.

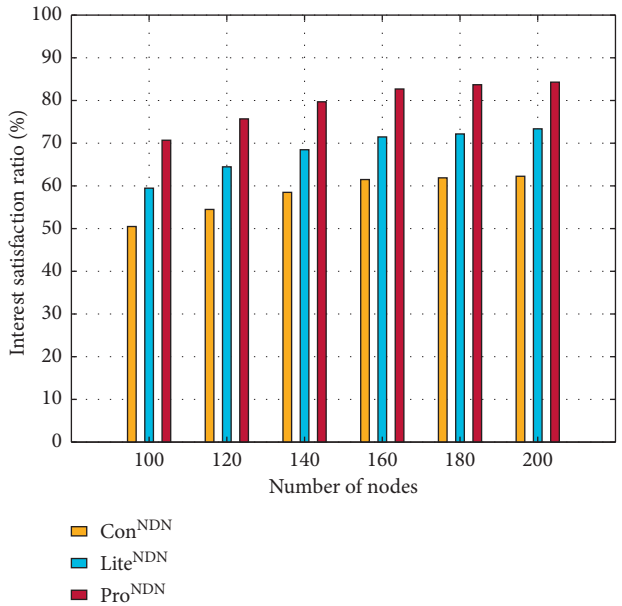


Figure 10: The performance of Interest satisfaction ratio against the number of nodes.

7. Conclusion and Future Work

In this paper, we proposed the Pro^{NDN}, a novel stateful forwarding and in-network caching strategy for NDN networks. The Pro^{NDN} consists of multicriteria decision-making (MCDM) based Interest forwarding and cooperative data caching. In the MCDM-based Interest forwarding, each outgoing interface alternative is first evaluated based on multiple network metrics to obtain the forwarding index, which is an indicator of the overall performance. Then, all outgoing interface alternatives are ranked in terms of the forwarding index, and the highest ranked one is chosen to forward the Interest packet. In addition, the cooperative data caching consists of two schemes: CacheData, which caches the data, and CacheFace, which caches the outgoing interface. For performance evaluation, we considered interface utilization ratio, round-trip time (RTT), and NACK ratio as real-time network metrics. We also developed a customized discrete event-driven simulation framework by using OMNeT++ and evaluated its performance through extensive simulation experiments. The simulation results show that the Pro^{NDN} can improve Interest satisfaction ratio and Interest satisfaction latency as well as reduce hop count and Content Store utilization ratio, indicating a viable stateful forwarding and in-network caching strategy in NDN networks.

As a future work, we plan to investigate analytic network process (ANP) to analyze the interrelationships between decision levels and multiple network metrics and dynamically calculate the relative weights of multiple network metrics. In addition, we plan to further extend the proposed MCDM-based Interest forwarding with the feature of Interest traffic load balancing. For example, the router can stochastically select an outgoing interface to forward the Interest packet by randomly generating a number and comparing it with the forwarding index of each outgoing interface. If the forwarding index of outgoing interface is larger than randomly generated number, this outgoing interface is chosen to forward the Interest packet. In this way, each outgoing interface will have a chance to forward the Interest packet, which can achieve the goal of traffic load balancing. In addition, in the context of Internet of Everything or 5G [27], as the number of connected devices significantly increases in the network, the amount of network metrics information that is used to make forwarding decision will be increased extensively as well. As a result, a longer computational latency of making forwarding decision could be observed at each intermediate router, which becomes a nontrivial problem. Thus, we plan to investigate the algorithm optimization of forwarding strategy to balance the tradeoff between algorithm efficiency and computational latency.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request considering the purpose of use in further research in the area.

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

This work was supported by startup grant in the Department of Computer Sciences and Electrical Engineering at Marshall University.

References

[1] C. Pu, "Sybil attack in RPL-based internet of things: analysis and defenses," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4937–4949, 2020.

[2] G. Price, "Cisco annual internet report (2018–2023) white paper," 2020.

[3] C. Pu, N. Payne, and J. Brown, "Self-adjusting share-Based countermeasure to interest flooding attack in named data networking," in *Proceedings of the IEEE Conference on Cyber, Physical and Social Computing*, pp. 142–147, Atlanta, GA, USA, July 2019.

[4] L. Zhang, A. Afanasyev, J. Burke et al., "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[5] Z. Li, Y. Xu, B. Zhang, L. Yan, and K. Liu, "Packet forwarding in named data networking requirements and survey of solutions," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, p. 1950, 1987.

[6] A. Afanasyev, "NFD developer's guide," Tech. Rep. NDN-0021, Department of Computer Science, University of California, Los Angeles, CA, USA, 2014.

[7] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A case for stateful forwarding plane," *Computer Communications*, vol. 36, no. 7, pp. 779–791, 2013.

[8] M. Abdelaal, M. Karadeniz, F. Dürr, and K. Rothermel, "liteNDN: QoS-aware packet forwarding and caching for

named data networks," in *Proceeding of the IEEE CCNC*, pp. 1–9, Las Vegas, NV, USA, January 2020.

[9] A. Varga, "OMNeT++," 2014, http://www.omnetpp.org/.

[10] A. Abane, M. Daoui, S. Bouzefrane, and P. Muhlethaler, "A lightweight forwarding strategy for named data networking in low-end IoT," *Journal of Network and Computer Applications*, vol. 148, Article ID 102445, 2019.

[11] Y. Zhang, B. Bai, K. Xu, and K. Lei, "IFS-RL: an intelligent forwarding strategy Based on reinforcement learning in named-data networking," in *Proceedings of the NetAI'18: Proceedings of the 2018 Workshop on Network Meets AI & ML*, pp. 54–59, New York, NY, USA, August 2018.

[12] P. Moll, J. Janda, and H. Hellwagner, "Adaptive forwarding of persistent Interests in named data networking," in *Proceedings of the ACM Conference on Information-Centric Networking (ICN 2017)*, pp. 180-181, Berlin Germany, September 2017.

[13] J. Yao, B. Yin, X. Tan, and X. Jiang, "A POMDP framework for forwarding mechanism in named data networking," *Computer Networks*, vol. 112, pp. 167–175, 2017.

[14] J. Lv, X. Tan, Y. Jin, and J. Zhu, "DRL-based forwarding strategy in named data networking," in *Proceedings of the 2018 37th Chinese Control Conference (CCC)*, pp. 6493–6498, IEEE, Wuhan, China, July 2018.

[15] R. Hou, L. Zhang, T. Wu, T. Mao, and J. Luo, "Bloom-filter-based request node collaboration caching for named data networking," *Cluster Computing*, vol. 22, no. 3, pp. 6681–6692, 2019.

[16] F. R. C. Araújo, A. M. de Sousa, and L. N. Sampaio, "SCaN-Mob: an opportunistic caching strategy to support producer mobility in named data wireless networking," *Computer Networks*, vol. 156, pp. 62–74, 2019.

[17] M. Yu and R. Li, "Dynamic popularity-Based caching permission strategy for named data networking," in *Proceedings of the 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 576–581, Nanjing. China, May 2018.

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

[19] H. Yan, D. Gao, W. Su, C. H. Foh, H. Zhang, and A. V. Vasilakos, "Caching strategy Based on hierarchical cluster for named data networking," *IEEE Access*, vol. 5, pp. 8433–8443, 2017.

[20] 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.

[21] G. Tzeng and J. Huang, *Multiple Attribute Decision Making: Methods and Applications*, CRC Press, Boca Raton, FL, USA, 2011.

[22] A. Afanasyev et al., "Packet fragmentation in NDN: Why NDN uses hop-by-hop fragmentation," Tech. Rep. NDN-0032, Department of Computer Science, University of California, Los Angeles, CA, USA, 2015.

[23] A. Medina, I. Matta, and J. Byers, "ndnSIM 2.0: a new version of the NDN simulator for NS-3," Tech. Rep. NDN-0028, Boston University, Boston, MA, USA, 2000.

[24] S. Tiennoy and C. Saivichit, "Using a distributed roadside unit for the data dissemination protocol in VANET with the named data architecture," *IEEE Access*, vol. 6, pp. 32612–32623, 2018.

[25] C. Yi, A. Afanasyev, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 3, pp. 62–67, 2012.

[26] C. Yi, "Adaptive forwarding in named data networking," Doctor Of Philosophy Dissertation, The University of Arizona, Tucson, Arizona, 2014.

[27] C. Pu and L. Carpenter, "$Psched$: a priority-Based service scheduling scheme for the Internet of drones," *IEEE Systems Journal*, pp. 1–10, 2020.