IEACC: An Intelligent Edge-Aided Congestion Control Scheme for Named Data Networking With Deep Reinforcement Learning

Jiayu Yang, *Student Member, IEEE*, Yuxin Chen, Kaiping Xue, *Senior Member, IEEE*, Jiangping Han, Jian Li, *Member, IEEE*, David S. L. Wei, *Life Senior Member, IEEE*, Qibin Sun, *Fellow, IEEE*, and Jun Lu

***Abstract*—As a promising implementation of Information-Centric Networking (ICN), Named Data Networking (NDN) has potential advantages over the TCP/IP network in content distribution, mobility support, etc. However, the research on NDN is still in its infancy, and congestion control, NDN's most important functional element, poses many challenges, such as congestion detection, excessive window reduction for non-congested paths, and unfairness. In this paper, we propose an Intelligent Edge-Aided Congestion Control (IEACC) scheme for the NDN network based on Deep Reinforcement Learning (DRL). The proposed IEACC provides a proactive congestion detector that utilizes intermediate routers to transmit accurate congestion information along the path to consumers through data packets. Furthermore, considering the multi-source transmission in NDN, IEACC divides data packets into different congestion degrees by a lightweight clustering algorithm and provides suitable inputs for DRL, thereby obtaining a reasonable transmission rate. Then, it distributes the estimated bandwidth resources to consumers with transmission needs to maintain fairness. Finally, we implement our proposed scheme in the simulation platform and evaluate the performance in different scenarios. The results show that it can improve data transmission rate, reduce packet loss, and maintain fairness compared with others.**

***Index Terms*—Named data networking, congestion control, reinforcement learning, fairness.**

I. Introduction

DUE to the fast-growing access demands of numerous mobile devices, data delivery volume on the Internet has risen dramatically in recent years. The vast mobile traffic has led to an emerging trend of addressing the mobility problem of the Internet. Meanwhile, the increase of video data, currently accounting for 66% of all mobile data traffic is expected to increase to 77% in 2026 [1], also intensifies the task of content distribution. They make host-based transmission a long-standing challenge in the existing Internet architecture.

Information-Centric Networking (ICN) [2] brings a new transmission mode to meet various distribution requirements of multiple network services and enhance mobility support. It changes the focus of the Internet architecture from the current "where" (location) to "what" (content). Named Data Networking (NDN), proposed by Zhang *et al.* [3], is one of the most typical ICN architectures, which identifies content by specific name and may cache content in original content repositories and caches of intermediate routers. Thus, consumers in NDN can obtain content from multiple content sources based on content name, which overcomes the pitfalls of end-to-end communications for mobile transmission and content distribution. NDN also provides two unique features for its multi-source transmission mode: the receiver-driven pull method and the one-interest-one-data principle. The former means that a receiver requests content by sending interest packets and the data source routers satisfying the requirements return the data. The latter restricts that one interest retrieves at most one data packet. Therefore, NDN controls the transmission rate of data packets by controlling the rate of interest packets.

Though the multi-source transmission mode has the potential for faster content distribution and great mobility support, it brings new challenges to congestion control. 1) **Congestion Detection**: As interest requests in NDN may be responded to through different sources with discrepant Round Trip Time (RTT), the estimation of Retransmission Time Out (RTO) is inaccurate, making RTT-based and RTO-based congestion detection unreliable. Moreover, there is no ACK mechanism, rendering the detection method utilizing duplicated-ACK invalid. Therefore, NDN can only depend on inaccurate RTOs to detect congestion with a high cost of massive packet losses. 2) **Congestion Adjustment**: The multi-source transmission mode increases the difficulty to predict data packets' responding location in NDN. Besides, the forwarding strategy [4] of interest requests further intensifies the complexity of data distribution. In this case, traditional congestion control methods may reduce the transmission efficiency when applied to NDN.

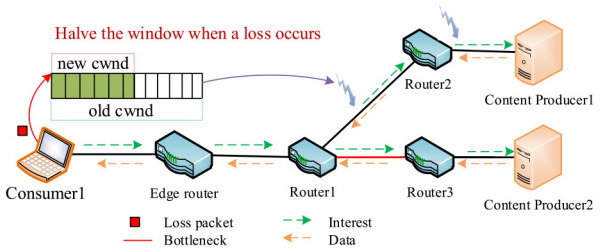
Manuscript received 21 December 2021; revised 6 April 2022 and 19 July 2022; accepted 22 July 2022. Date of publication 4 August 2022; date of current version 31 January 2023. This work is supported in part by the National Natural Science Foundation of China under Grants No. 61972371 and No. U19B2023, and Youth Innovation Promotion Association of the Chinese Academy of Sciences (CAS) under Grant No. Y202093. The associate editor coordinating the review of this article and approving it for publication was A. Detti. *(Corresponding author: Kaiping Xue.)*

Jiayu Yang, Yuxin Chen, Kaiping Xue, Jiangping Han, Jian Li, and Qibin Sun are with the School of Cyber Science and Technology, University of Science and Technology of China, Hefei 230027, Anhui, China (e-mail: kpxue@ustc.edu.cn).

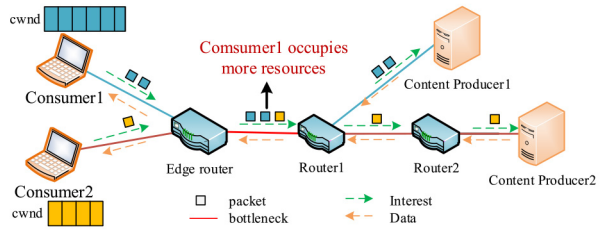
David S. L. Wei is with the Computer and Information Science Department, Fordham University, Bronx, NY 10458 USA.

Jun Lu is with the School of Cyber Science and Technology and the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, Anhui, China.

Digital Object Identifier 10.1109/TNSM.2022.3196344



(a) The excessive traffic reduction of non-congested paths.



(b) The consumer requesting popular content occupies more bandwidth.

Fig. 1. The challenges of congestion control in NDN.

As shown in Fig. 1(a), when *Consumer1* treats data from two different sources uniformly, it will halve the window if it senses a loss packet to *Producer2*, which causes unnecessary traffic decrease for non-congested paths to *Producer1*.

Fairness is also one of the urgent issues to be solved in NDN. Padhye *et al.* [5] proposed that competing flows attain throughputs inversely proportional to their RTTs. In NDN, popular content is usually located closer to consumers with the existence of in-network caches and can be obtained with less delay. Thus, if a consumer grows its sending window when it receives data packets according to the predefined rules, the consumer that requests for popular content may occupy the bandwidth of others, which leads to unfairness problems [6]. As shown in Fig. 1(b), *Consumer1* requesting popular content that returns with smaller RTTs may occupy more bandwidth resources at the bottleneck compared with *Consumer2*.

The traditional receiver-based methods usually utilize heuristic algorithms that adjust the congestion window size following static pre-defined policies inherited from Transmission Control Protocol (TCP) [7], [8]. Most existing works think that distinguishing multiple paths can solve the above problem. They proposed to add Path Tag to data packets to identify alternative paths and to execute congestion control similar to TCP flows [9], [10]. As the location of the content is changing with caching strategies and complex requests in the network, fixing forwarding paths may limit the flexibility of NDN content acquisition. Other work implements congestion control at intermediate nodes that can obtain more accurate congestion information. However, they usually have strong assumptions about known path bandwidth [11], [12], [13], or need to maintain a considerable amount of information and utilize the forward strategy to adjust the traffic load of each interface [14], [15].

Therefore, we give up the strong assumptions about forwarding and caching strategies of the NDN network and retain flexible content acquisition of NDN. We deploy the congestion control module at the consumer (or edge router), which adaptively adjusts the transmission rate depending on the network status of the in-network. However, designing the rate control scheme that adapts to the complex and changeable responding locations in NDN is challenging. The traditional methods using pre-defined strategies are powerless and may reduce transmission efficiency. Thus, we solve this problem using emerging new technologies, in which Deep Reinforcement Learning (DRL) can process a large amount of data, mine information from its experience data, and learn the optimal strategy itself. Lan *et al.* [16] already utilized DRL in their scheme DRL-CCP. They provided a limited discrete action space and leveraged RTT as the congestion signal, which leads to misjudgments. Besides, they implemented the DRL model in each consumer, which causes a huge overhead.

In this paper, we propose an Intelligent Edge-Aided Congestion Control (IEACC) scheme consisting of three modules: Proactive Congestion Detector (PCD), Intelligent Rate Adjustment (IRA), and Fair Resource Allocation (FRA). Specifically, PCD is a proactive congestion notification strategy that monitors data packet queue length at intermediate routers to feedback accurate congestion information to consumers through data packets. Furthermore, to reduce the deployment overhead of DRL and maintain fairness, IEACC implements the DRL agent for rate control at the edge router. It leverages IRA to adjust the rate of interest packets injected into the in-network through DRL. Then, it utilizes FRA to maintain fairness by allocating the estimated transmission resources by DRL to consumers, who regularly update their sending rates accordingly.

The main contributions of this paper are summarized as follows:

- We propose an intelligent edge-aided congestion control scheme called IEACC for the NDN network, which achieves adaptive congestion control by DRL. The state, action, and reward functions of IEACC are specially designed based on the unique data acquisition mode of NDN. As deployed at the edge, one DRL model can serve multiple consumers, which reduces the deployment overhead of DRL in the NDN network.
- We further provide a proactive congestion detector, which enables intermediate routers to sense and feedback accurate local congestion information through data packets. Then, the edge router can derive the congestion status of the global network from the returned information by a lightweight clustering algorithm, which improves the DRL performance.
- We implement the IEACC scheme in the simulation platform and evaluate the performance in different scenarios. For communication between the DRL agent and NDN simulated hosts, we extend the original ndnSIM with a data processing method and interactive interfaces. In both static and dynamic network environments, the results reveal that IEACC can improve the data transmission rate, up to 100.82% and 34.84% compared to ICP and DRL-CCP, respectively.

The rest of this paper is organized as follows. In Section II, we present the background of NDN, RL, and our motivation.

The related work and our solution are illustrated in Section III and Section IV, respectively. Subsequently, we give performance evaluation in Section V. Finally, we conclude our work in Section VI.

II. Background and Motivation

In this section, we firstly provide background knowledge of NDN network, Reinforcement Learning (RL), and then present our motivation.

A. NDN Network

NDN was first proposed by Jacobson *et al.* [17] and then attracted attention from many researchers and got support via NSF projects [3]. It redesigns the network architecture and changes the end-to-end transmission mode of the traditional network to a content-centric network identified by specific content name. There are two kinds of packets in NDN: interest and data, and each NDN router has three types of data structures: Forwarding Information Base (FIB), Content Store (CS), and Pending Interest Table (PIT). The basic communication process of NDN is presented as follows: A consumer sends out an interest packet carrying the name that identifies the requested data. When a NDN router receives an interest packet, it searches its CS and sends the data back to consumers if it has the data packet matching the request. Otherwise, it checks the PIT table and adds it to the corresponding entry if there already exists the request, or else it forwards the interest request according to the FIB. Correspondingly, when a data packet returns to a router, it checks the PIT table and sends the data to the interfaces with the request. Otherwise, it drops the packet. In the transmission of data packets, routers can decide whether to cache the content to their CSs according to cache strategies [18]. Thus, some interest requests can be flexibly responded to by multiple resources including original content repositories and intermediate routers.

The objective of congestion control in NDN is, the same as in traditional networks, to avoid packet loss during transmission and maximize network utilization. NDN provides the receiver-driven "pull" mode and one-interest-one-data principles for congestion control, which restricts that one interest packet at most receives one data packet. Thus, in NDN, congestion control algorithms adjust the number of interest packets sent by consumers to control the number of data packets transmitted in the network.

B. Deep Reinforcement Learning

The RL problem is formalized as a discrete time stochastic control process, in which the agent interacts with its environment in the following way: at each discrete decision epoch *t*, the agent observes the state s_t of the environment in the state set $\mathcal{S}$, and then selects the act a_t from action set $\mathcal{A}$ according to its policies and gets the corresponding reward r_t. Thus, the main task of an agent is to find a policy $\mu(s)$ mapping a state to an action, $\mu(s) : \mathcal{S} \rightarrow \mathcal{A}$ (deterministic), or denotes the probability that action a may be chosen in the state s, $\mu(s, a) : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ (stochastic), to maximize the total future reward

$$\mathcal{R}_t = r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t} r_T, \tag{1}$$

where *T* is the last decision time, and $\gamma \in (0, 1]$ is the discount factor.

The DRL was introduced by Mnih *et al.* [19], who extend the well-known Q-Learning using tables to represent $\mathcal{Q}(s, a)$ to Deep Q-Network (DQN) utilizing a DNN to derive the correlation between each state-action pair (s_t, a_t). When training the neural network, DRL optimizes a loss function. That is, the goal of DRL is to minimize the loss function (the deviation between the target and network output):

$$L(\theta^{\mathcal{Q}}) = \mathbb{E}\Big[y_t - \mathcal{Q}\Big(s_t, a_t|\theta^{\mathcal{Q}}\Big)\Big]. \tag{2}$$

Where y_t is the target value of training for each state-action pair derived from the Bellman equation

$$y_t = r(s_t, a_t) + \gamma \mathcal{Q}\Big(s_{t+1}, \mu(s_{t+1})|\theta^{\mathcal{Q}}\Big). \tag{3}$$

Although DQN performs well in low-dimensional observation spaces with DNN, it is still incapable of congestion control in NDN that needs continuous control. In recent studies, Lillicrap *et al.* [20] introduced an actor-critic approach based on DNN and the deterministic policy gradient [21] called Deep Deterministic Policy Gradient (DDPG) for continuous control. Specifically, DDPG maintains two DNNs: The actor part is used to approximate the policy function $\mu(s_t|\theta^{\mu})$ and generate actions to interact with the environment. The critic part is responsible for approximating the value function $\mathcal{Q}(s_t, a_t|\theta^{\mathcal{Q}})$ and evaluating the performance of the actor in each interaction cycle.

C. Motivation

Firstly, we analyze how congestion occurs in NDN networks. In NDN, there is no fixed end-to-end connection because of its multi-source transmission mode. However, the cause of congestion in NDN is still the same as that of TCP/IP networks. If consumers send too many interest requests, returned data packets may exceed the link bandwidth and queue in the transmission buffer. When the queue length exceeds the buffer capacity, it causes packet loss, then congestion occurs. It reveals why traditional schemes that adopt TCP-like congestion control algorithms still make sense in NDN networks. However, the cost of using packet loss as a congestion detector is expensive because consumers can only sense packet loss through RTO in NDN. Therefore, we enable the intermediate routers to monitor data packet queue length for accurate congestion detection and explicitly inform consumers by carrying information in data packets.

Although with accurate congestion information, suitable congestion control is still some way off. Specifically, data packets may come from different content storing locations, which are constantly changing according to cache strategies [18] and cache replacement schemes [22]. The traditional methods using pre-defined strategies are powerless to adapt to changeable responding locations in NDN and may reduce transmission efficiency. Therefore, we leverage the emerging technology DRL that can mine information and learn the

optimal strategy by itself. However, deploying DRL on each consumer may have a higher cost, so we need new designs. In this paper, we propose to implement DRL at edge routers, which can not only avoid congestion by adjusting the traffic flowing into the in-network but also be a trusted neutral to maintain fairness. Firstly, with the development of equipment, NDN routers may have higher computing power than that of ordinary routers. The edge router, because of its special location (connecting users and intranets), are considered to have higher processing capabilities. There are many articles that perform more complex processing at the edge [23], [24], [25]. It is reasonable for its computing power to support the DRL application in IEACC. There are also other advantages: the edge router with more comprehensive transmission information from multiple consumers is more suitable to train and update the DRL model and improve rate control efficiency. It can reduce deployment overhead as the number of edge routers is far less than that of consumers. Besides, edge-aided congestion control can integrate with multiple NDN security policies, such as access control [23] and content integrity authentication [24], to form a more secure and effective NDN network.

The usual way to implement congestion control is to maintain a congestion window, which grows and decreases as data packets are returned and lost. Packets with low transmission delay come back faster. Thus, when transmission delays of paths are different, their window growth rates are different, which causes unfairness. In this paper, we avoid the imbalance caused by transmission delays by allocating transmission rates to consumers instead of maintaining a window. Consumers with the same window size may send different data packets within a certain period of time if they have disparate transmission delays, however, the same transmission rate does not.

As for the selection of the DRL algorithms, we consider both training difficulty and data processing ability. The first algorithm that comes to mind is DQN [19], which is the most common algorithm of DRL. It has low training difficulty and converges rapidly for utilizing only one neural network. However, it is only suitable for low dimension data inputs and can only output discrete action values in the predefined action space. To achieve fine-grained rate adjustment in NDN networks, we need continuous control algorithms that can choose any value in the action space according to the status of the environment. As far as we know, Deep Deterministic Policy Gradient (DDPG) [20], Asynchronous Advantage Actor-Critic (A3C) [26] and Twin Delayed Deep Deterministic Policy Gradient (TD3) [27] are commonly used algorithms for continuous control. These algorithms are all suitable for transmission rate control in the NDN network, though, with different training overhead. Both A3C and TD3 are more difficult to converge for having more than two neural networks. However, the DDPG algorithm with only two DNNs has lower training difficulty and good stability. Thus, we adopt DDPG to implement rate control in our scheme.

III. Related Work

The congestion control mechanism is one of the key elements of NDN, which has attracted much attention from researchers. According to the control mode, the existing works of congestion control in NDN are classified into three categories: receiver-based control, hop-by-hop interesting shaping, and hybrid method [6].

The receiver-based mechanism consists of single-source and multi-source algorithms, which maintains one or more congestion windows at the consumer side to restrict the number of interest packets injected into networks. One of the typical single-source algorithms is Interest Control Protocol (ICP) [7], which utilizes the Additive Increase and Multiplicative Decrease (AIMD) algorithm for window adjustment. Others provided extensions of ICP [8], [28], they still have many congestion misjudgments for relying on RTT. Ren *et al.* [29] proposed an Explicit Control Protocol (ECP), which monitors the queue status at intermediate routers, and sets fixed thresholds to distinguish different congestion levels (Free, Busy, and Congestion). However, single-source algorithms are incompatible with the multi-source transport characteristic, which may cause unnecessary traffic reduction of non-congested paths and adjust window size based on inaccurate RTOs. Some works maintain a separate window for each source [30], each path [31], or each Content Store (CS) [32]. Ye *et al.* [9] proposed Path-specified Transport Protocol (PTP), which introduces a Path Tag in the data packet to record the interface identification of each router along the path to destination sources. Thus, consumers can maintain a separate window for each path to avoid excessive reduction on non-congested paths. Wu *et al.* [10] also proposed a multi-path discovery and multi-path congestion control scheme, named MPCC, which utilizes Path Tag in both interest and data packet to discover new paths and record path information. Then, it utilizes the Upper Confidence Bound algorithm to select sub-paths and maximize the network throughput. However, these algorithms may restrict the flexible content acquisition in NDN. There is also a scheme that utilizes DRL technology [16], which trains a DQN model to make finer window adjustments. However, it ignores the multi-source characteristic in the NDN network, still utilizes inaccurate RTTs as congestion notifications. Besides, it can not maintain fairness in NDN.

The hop-by-hop algorithms implement congestion control at intermediate routers, which adjust the interest forwarding rate or transfer data traffic from the congested interface to free ones. Some of them utilized strong assumptions about knowing path bandwidth to adjust transmission rate at intermediate routers [11], [12], [13]. Schneider *et al.* [15] provided Practical Congestion Control (PCON) scheme, which detects congestion based on the CoDel AQM and marks certain packets to explicitly transmit congestion information. The downstream routers divert traffic to alternative interfaces to avoid congestion, and consumers can reduce their interest sending rates based on marked packets. Nikzad *et al.* [33] proposed a Responsibility-based Transport Control (RTC) method, where

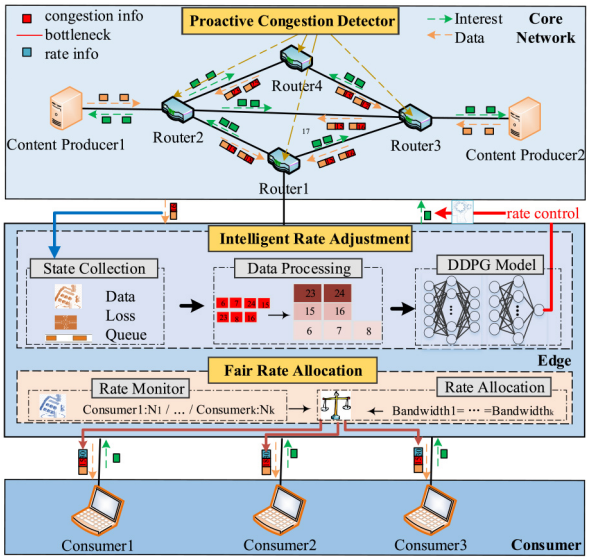


Fig. 2. The architecture of the proposed scheme.

Interest packet	Data packet
Content Name	Content Name
Selector (order preference, publisher filter, Scope, …)	Signature (digest algorithm, witness,…)
	Signed info (publisher ID, key locator, stale time, …)
	Data
Nonce	Congestion Info
Later Prefix	Allocated Rate

Fig. 3. The new format of interest and data packet.

the intermediate routers can decide on accepting or refusing to take responsibility for forwarding a newly received interest packet. In [14], the Coordinate Congestion Control Protocol (3CP) proposed by Hashemi *et al.* decouples forwarding and congestion control plane. It adds several new formats in interest and data packets to transmit congestion information and estimated resources along the path. It adjusts the forwarding probability to certain interfaces and consumer's sending windows according to the positive and negative signals conveyed by data packets. Although these algorithms utilize the complex operations at intermediate routers to obtain more accurate transmission rates, their extremely high overhead for rate control at each intermediate router cannot be ignored. Also, their approach is extremely difficult to deploy.

The hybrid method combines the former two methods, which implements both interest shaping at intermediate routers and window adjustment at consumers [34]. The more precise rate control may lead to better efficiency, which is also complicated and introduces huge computational burdens to the network. Besides, there are possible conflicts between consumers and intermediate routers when using two rate control methods together.

Our solution can be classified as the receiver-based method, which maintains a rate value for a interface at the edge router and does not make any assumptions about caching and forwarding strategies at intermediate routers.

IV. Intelligent Edge-Aided Congestion Control

In this section, we introduce our Intelligent Edge-Aided Congestion Control (IEACC) scheme, which is proposed for NDN to cope with the challenges in inaccurate congestion detection, excessive window reduction, and consumer unfairness. We present the design details in the following.

A. Design Overview

Firstly, we provide a design overview to present an overall idea of the scheme. As shown in Fig. 2, IEACC consists of three modules: Proactive Congestion Detector (PCD), Intelligent Rate Adjustment (IRA), and Fair Rate Allocation (FRA). The PCD enables intermediate routers of NDN to proactively monitor the queue length of data packets. For each passing data packet, it updates the current maximum queue length value and conveys the congestion information to consumers. The IRA and FRA methods are both implemented at the edge router. Specifically, IRA collects path information within the in-network, classifies data according to their congestion degrees to provide suitable inputs for DRL, and outputs a reasonable transmission rate for interest packets. Then, FRA allocates the estimated transmission resources to consumers with requirements. It calculates the updated transmission rate on the basis of ensuring the consistent transmission bandwidth of each consumer, which is then conveyed to consumers through data packets. Thus, the consumers detect new transmission rates carried in data packets and send interest requests accordingly. Therefore, with the accurate congestion information delivered by the intermediate routers, IEACC uses DRL to control the interest packets injected into the core network and updates consumers' transmission rate according to the principle of bandwidth consistency. It can not only improve network transmission efficiency but also maintain fairness among consumers.

We present the detailed descriptions of the three proposed algorithms in the following sub-sections.

B. Packets Format

To transmit auxiliary control information between NDN nodes with little cost, we design new packet formats as shown in Fig. 3. The functions of new adding fields are elaborated as follows:

- Later Prefix: Prefix is an important components of content name, which is the basis for caching and routing decisions in NDN networks and is useful for providing prediction of future traffic. Each consumer sets this field by its later interest requests, which is transmitted to the edge router and then will be deleted before being transmitted to the core network to reduce transmission cost.
- Congestion Info: Each intermediate router modifies this field to transmit the maximal congestion degree to consumers.
- Allocated Rate: The edge router updates this field according to the rate allocation strategy to deliver new rate to consumers.

C. Proactive Congestion Detector

In the original implementation of NDN, NACK [35] is usually used as a signal of congestion conditions, which is triggered when queue length exceeds the predefined threshold. The consumers who receive a NACK packet will slow down their sending rate to avoid packet losses. Although NACK can convey congestion information, it can only transmit congestion signals of "yes" or "no" and is unable to transmit specific congestion conditions, which has a limited effect.

To get accurate congestion information, IEACC provides a proactive congestion detector, which transmits explicit congestion information to consumers through data packet with little cost. There are many options that can be used as congestion signals including RTT, data packet queue length, interest packet queue length, and the transmission delay of adjacent routers. Among them, we choose the queue length of the data packet because data packets are the main cause of traffic congestion. Besides, the queue length is not affected by transmission delay.

As shown in Fig. 3, IEACC appends an additional "Congestion Info" field in the original data packet to convey the queue information of intermediate routers along the path. Although it is able to carry the congestion status of each intermediate node for more fine-grained analysis, the design also introduces more overhead, especially when the path is long. Thus, IEACC conveys only the maximal queue length that indicates the highest congestion degree along the path to consumers. Specifically, the appended field is 16-bit and can convey the queue length value smaller than 65535, which is sufficient for the existing implementation in ndnSIM [36]. For more accurate control, each intermediate router timely updates its queue length when a data packet arrives or is forwarded to obtain instant congestion information. Before forwarding the data packets, each router compares the current queue length with the stored value in the data packet and rewrites the stored value if it has a higher congestion level.

D. Intelligent Rate Adjustment

With accurate congestion information, the edge router utilizes DRL method to obtain a reasonable transmission rate by the IRA module. Specifically, IRA utilizes data from multiple interfaces of the edge router to train the model and outputs the updated transmission rate for each interface through its input. In this case, the model is decoupled from the network topology and can be flexibly applied to various topological environments. As shown in Fig. 4, a DRL agent is composed of the state, action, and reward, which are elaborated as follows:

State: A reasonable state design can not only accelerate convergence but also reduce the training complexity and overhead of the DRL model.

As mentioned before, the congestion information carried in data packets is the local information of each path. However, data in NDN networks may go through different paths with disparate congestion states to reach the edge router. To achieve a more accurate transmission rate, we need global network status. Thus, IRA contains a lightweight clustering algorithm for data analysis, which further provides more accurate inputs for

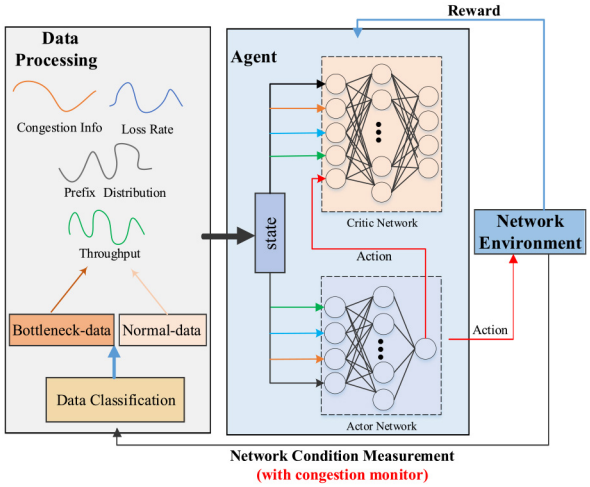


Fig. 4. The network of the DDPG model.

Algorithm 1: Data Processing

Input: The DRL agent updating interval T, congestion level set $q = []$, data collection time $t = 0$, initial group number $n = 2$, classification parameter $\theta = 0.25$;

1 **while** $t \leq T$ **do**
2 for coming data, append the congestion level to q;
3 t grows with the passing time;
4 **end**
5 $g_1 = \min\{q\}$, $g_2 = \max\{q\}$;
6 **for** $i \leq lens\{q\}$ **do**
7 $d_{n,i} = \min\{\| q_i - g_1 \|, \ldots, \| q_i - g_n \|\}$;
8 **end**
9 $l = \min\{\overline{q} - g_1, g_2 - \overline{q}\}$;
10 **if** $\max\{d_n\} \geq l \cdot \theta$ **then**
11 $n++$;
12 $g_n = q_k$ $(d_{n,k} = \max\{d_n\})$;
13 $l = \min\{l, \| g_n - \overline{q} \|\}$;
14 go to the line 6;
15 **end**
16 **for** $i \leq lens\{q\}$ **do**
17 **for** $j \leq n$ **do**
18 $L_{i,j} = \| q_i - g_j \|$;
19 **end**
20 q_i belongs to category k, $L_{i,k} = \min\{L_{i,1}, \ldots, L_{i,n}\}$
21 **end**

DRL. Specifically, it introduces the monitoring interval to collect data and separates them by the interval to maintain Markov property. Then, it utilizes a moderately modified Max-Min-distance algorithm to cluster received data with acceptable computational overhead. The specific processing is shown in Algorithm 1.

Suppose that, in a data processing interval T, the maximum data queue length carried by the ith packet is q_i. IRA first selects $g_1 = \min\{q_1, \ldots, q_N\}$ and $g_2 = \max\{q_1, \ldots, q_N\}$ as the center points of the first two categories, and calculates $\bar{q}$, the average value of q. Then, it computes the distance from the

remaining data congestion levels to g_1 and g_2, and generates a new classification according to the following formula:

$$\max_i [\min [d_{i,1}, d_{i,2}]] \geq l \cdot \theta, i = 1, 2, \ldots, N, \tag{4}$$

where θ is the classification parameter, $l \cdot \theta$ is the boundary of new category, and

$$d_{i,j} = \|q_i - g_j\|, j = 1, 2, \ \ i = 1, 2, \ldots, N.$$
$$l = \min\{\bar{q} - g_1, g_2 - \bar{q}\}.$$

Then, it repeats the above formula until no new centers are generated. After confirming the number of categories and the cluster centers, the data can be sorted into various categories according to the principle that it has the minimum distance to the belonging cluster center as illustrated from Line 16 to Line 20. So the final number of categories is related to the distances among the received data, however, the state dimension of the reinforcement learning model is a certain number. Thus, IRA takes the classification of g_2 that has the maximum average queue length (congestion level) as the first category namely "bottleneck data", and the others as the second category namely "normal data".

Specifically, for each interface of the edge router, its input state at the specific epoch *t* is $S_t = [Gb_t, Gn_t, Q_t^{avg}, L_t, P_t]$, where Gb_t and Gn_t denote the goodput of "bottleneck data" and "normal data", respectively. Q_t^{avg} is the average value of queue length obtained from the PCD. L_t represents the number of lost packets in the epoch *t*, which may include interest and data packets notified by RTO. That is, if a sent interest packet does not return the corresponding data packet within the RTO time, a packet loss has occurred, and the value of L_t is increased by 1. We use the default RTO update algorithm and periodically transmit packet loss information from consumers to the edge router. P_t denotes the distribution of subsequent interest requests obtained from the "Later Prefix" contained in interest packets. Specifically, $P_t = [p^1, p^2, \ldots, p^h]$ is a one-dimensional vector that denotes the subsequent distribution of interest requests. Where p^i represents the request proportion of the *i*th popular content to all requests. The process of obtaining P_t is as follows: when the edge router receives an interest packet from consumers, it gets the prefix of later requests from the "Later Prefix" field. Then, it records the number of requests for various types of contents from consumers in the monitor interval to obtain the popularity prediction of local contents. At the start of the next interval, the edge router calculates P_t and inputs the prediction information into the model. The value of *h* can be adjusted with requirements, and we provide overhead and efficiency tests for several *h* values in Section V for reference.

It's worth noting that though we consider only two categories of data in our design, it can be easily extended to the scenario with more categories of data for fine-grained control by modifying the input dimension of DDPG model.

Action: The existing congestion control algorithms, such as ICP, ECP, and DRL-CCP, mainly control the sending rate of interest packets by adjusting cwnd size. They only consider discrete action space, which has a limited effect on rate control that is continuous. However, the DDPG algorithm in our DRL model can output continuous action through a neural network. As for the threshold setting of the action space, we refer to TCP algorithms in the traditional network. As far as we know, most loss-based TCP algorithms including NewReno [37], Cubic [38], and SCTP [39] follow a principle that the cwnd variation does not exceed exponential change. Specifically, the window grows up to twice the original size in the best case, and it is halved when a packet loss occurs. Thus, we provide continuous action space from 0.5 to 2, that is, the action is $a \in [0.5, 2]$. It is able to avoid great performance fluctuations. Thus, transmission rate in the epoch *t* can be calculated by

$$Rate_t = Rate_{t-1} \cdot a. \tag{5}$$

Reward: The design of reward is consistent with the objective of maximization network utility. It contains goodput (G_t) representing the current capacity of the interface, the loss value (L_t), and average queue length (Q_t^{avg}) indicating the congestion degree of transmission states. Specifically, the reward function is:

$$Reward_t = \alpha \cdot \log(G_t) - \beta \cdot Q_t^{avg} - \gamma \cdot L_t, \tag{6}$$

where

$$G_t = 1 + \frac{D_t \cdot N_t \cdot 8}{d \cdot 1024 \cdot 1024}, \tag{7}$$

and

$$Q_t^{avg} = \left(\sum_{i=1}^{N_t} Q_i\right) / N_t, \tag{8}$$

in which N_t is the number of received data packets in the epoch *t* that is composed of "bottleneck data" and "normal data", and *d* is the action interval, which is 0.2 in our design. D_t denotes the average size of received valid data packets. The computation of the Goodput is done by Eq. (7), where 8 and 1024 are to complete unit conversion. The computation of average queue (Q_t^{avg}) is shown in Eq. (8), in which Q_i denotes the maximum data queue length carried by the *i*th packet, and N_t is the total number of received data packets in the epoch *t*. In Eq. (6), α, β, and γ are the parameters that represent the importance of Goodput, queue length, and packet loss, respectively. These parameters are all positive, which means that the reward expects to increase the goodput, but at the same time to reduce the average queue length and packet loss. The exact values of these parameters have a significant influence on the presentation of the model. We do a lot of tests and finally set $\alpha = 0.9$, $\beta = 1/32$, and $\gamma = 1/100$.

Implementation: In this part, we mainly introduce the detail of neural network design and some useful optimization strategies for algorithm operation. For the design of actor and critic networks, IRA utilizes three-layer fully-connected feed-forward neural networks. It introduces non-linear function approximators into the network to explore more complex data relations in NDN, where there are 30 neurons in each layer with the Leaky Rectifier for activation. The critic network can be trained as a regular DQN, which outputs the corresponding Q-value for a given action-state pair using

$$\mathcal{Q}^{\pi}(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}[r(s_t, a_t) + \gamma \mathbb{E}[\mathcal{Q}^{\pi}(s_{t+1}, a_{t+1})]].$$

When the target policy is deterministic $\mu : \mathrm{S} \leftarrow \mathcal{A}$, the inner expectation can be avoided. Thus, the above equation can be written as:

$$\mathcal{Q}^{\mu}(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}[r(s_t, a_t) + \gamma \mathcal{Q}^{\mu}(s_{t+1}, a_{t+1})].$$

To learn parameter μ, the updating algorithm based on the gradient is usually adopted [40]. The main step is to calculate the gradient of the sum of expectations. For the actor network, DDPG maintains a parameterized actor function $\mu(s|\theta^{\mu})$ that specifies the current policy by deterministically mapping states to a specific action. The chain rule is applied to the expected cumulative reward J considering the actor parameters:

$$\begin{aligned}\nabla_{\theta^{\mu}} \mathcal{J} &\approx \mathbb{E}\Big[\nabla_{\theta^{\mu}} \mathcal{Q}\Big(s, a|\theta^{\mathcal{Q}}\Big)|_{s=s_t, a=\mu(s_t|\theta^{\mu})}\Big] \\ &= \mathbb{E}\Big[\nabla_{a} \mathcal{Q}\Big(s, a|\theta^{\mathcal{Q}}\Big)|_{s=s_t, a=\mu(s_t)} \cdot \nabla_{\theta^{\mu}} \mu\big(s, a|\theta^{\mu}\big)|_{s=s_t}\Big].\end{aligned}$$

In the training, IRA utilizes the replay buffer and target network techniques to ensure the convergence of approximate functions as DQN. It also gains more benefit from a set of uncorrelated transitions and utilizes batch training. The update mode is changed to soft updates as

$$\begin{aligned}\theta^{Q'} &\leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}, \\ \theta^{\mu'} &\leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'},\end{aligned}$$

for the target actor and critic networks, rather than directly copy the weights used in DQN. We also normalize the input value included in the state to eliminate the magnitude difference among various types of data, such as Gb_t and Gn_t may reach ten thousand while L_t is much smaller.

E. Fair Resource Allocation

The fairness problem of the NDN network is still an open issue due to the multi-source transmission mode, various content requests from consumers, and uncertain access paths. We provide a preliminary algorithm, which is committed to providing fairness for consumers that are connected to the same edge. We leave other complex cases as our future work.

Mainly, we achieve fairness through resource allocation strategies, which include centralized decision-making and decentralized decision-making [41]. The main difference between the two types is whether the executor of resource allocation is the resource owner or the consumer (edge). IEACC utilizes the latter method to adjust transmission rate according to consumers' transmission requirements and congestion feedback from in-network through the IRA module. Thus, the FRA module introduces a simple and effective centralized decision-making strategy to fairly allocate the estimated bandwidth to consumers. However, consumers may have different transmission delays to the edge router, which leads to unfairness if they independently increase window size when data packets return. Thus, the edge router allocates a transmission rate to each consumer to avoid the effect of consumer-to-edge delay and ensure fairness.

Suppose that there are m consumers connected to the edge router E with n interfaces, where the transmission rate of interface i after the last rate adjustment of DDPG model is x_i. Then, edge router conveys the rate

$$R_{alloc} = \frac{\sum_{k=0}^{k=n} x_k}{m} + \frac{\eta \cdot Buffer - ocbuff}{d \cdot m}$$

to each consumer connected to it through the field of "Allocated Rate". The first part is the equal share of the transmission rate of each interface, and the second part is the basic rate provided by the edge buffer to prevent bandwidth underutilization when some interest packets only go to specific interfaces. *Buffer* denotes the total interest queue buffer of the edge router, *ocbuff* is the occupied buffer size, and d is the action interval. The parameter $\eta \in (0, 1)$ adjusts the percentage of buffer size used for containing excessive interest requests from consumers. In general, buffer queue occupation is a reasonable signal to reveal the congestion level of routers [42]. The setting of queue occupancy is closely related to transmission efficiency. Specifically, ensuring 100% buffer occupancy can improve bandwidth utilization, however, it often results in packet losses and increased queuing delay of data packets. If the router guarantees low queue occupancy (close to zero), there will be no packet loss, and the transmission delay also reduces. Nevertheless, in this case, the bandwidth utilization is hard to guarantee. Therefore, we choose the occupancy value of 50% to achieve a trade-off. To some extent, it can ensure bandwidth utilization, avoid severe packet losses, and reduces excessive queuing delay. Thus, in our experiment, we set $\eta = 0.5$.

Besides, to prevent severe fluctuations in the quality of service at consumer side, IEACC also provides upper and lower thresholds when consumers modify the rate to the specified value after receiving a new rate. As presented in the Eq. (9), a consumer with a current rate R_{old} receives a new allocated rate R_{alloc}, then it firstly compares whether R_{alloc} is larger than R_{old}. If so, it sets the new rate to be no more than twice of the original rate and takes the new allocated rate R_{alloc} as the upper bound. Otherwise, it ensures the new rate to be no less than half of the original rate and takes the new allocated rate R_{alloc} as the lower bound.

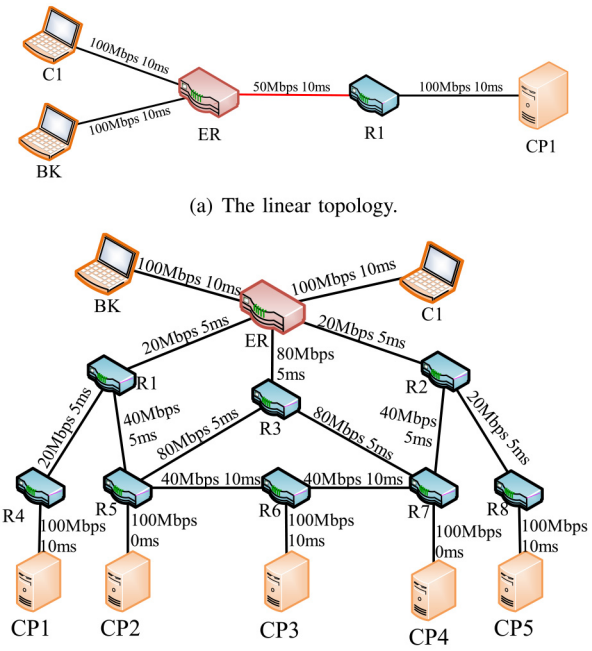
$$R_{new} = \begin{cases} \min\{R_{alloc}, 2 \cdot R_{old}\}, & if\ R_{alloc} > R_{old} \\ \max\{R_{alloc}, \frac{1}{2} \cdot R_{old}\}, & if\ R_{alloc} < R_{old} \end{cases} \quad (9)$$

It needs to note that, edge routers can also monitor the number of interest packets sent by each consumer to detect greedy consumers that do not obey the defined protocol. If a greedy consumer is found, the edge router can discard its interest requests to implement punishment. Beside, if resources are sufficient, the DRL model can also be deployed on each consumer, which can achieve fairness when consumers share the same bottleneck.

V. Performance Evaluation

We evaluate the performance of our proposed scheme IEACC via a simulation environment based on ndnSIM (version 2.7, an ns-3 based NDN simulator) [36] under the Ubuntu 18.04 virtual machine with 4 cores and 8GB memory. The receiver-based algorithms with which we compared include ICP [7], ECP [29], and DRL-based algorithm DRL-CCP [16]

using DQN. Specifically, ICP is the typical receive-based congestion control algorithm adjusting cwnd in an AIMD mode, and ECP is a control method based on explicit feedback that utilizes NACK to notify congestion. DRL-CCP is a method that maintains a single window and utilizes a discrete action set to perform window adjustment by DRL. Besides, we also compare IEACC with the popular hop-by-hop congestion control algorithm PCON [15]. Because IEACC has no load balancing measures at intermediate routers, the comparison is limited to specific scenarios for fairness. We measure the transmission efficiency and reliability of algorithms through data completion time (with specific transmission requirements), average transmission delay, and packet loss. It is worth noting that the packet loss of algorithms except ECP refers to the loss of data packets, while the ECP algorithm includes a part of interest packets. Although the loss of interest packets has a lower cost than data packets, it also causes other problems. For example, consumers can only know lost packet by waiting for RTOs. It will increase the out-of-order degree of data and the time of delivering data to the application layer (data must be complete before uploading). Moreover, we also present fairness performance by comparing the data transmission rate of each consumer.



(a) The linear topology.

(b) The complex topology.

Fig. 5. The topologies of performance testing.

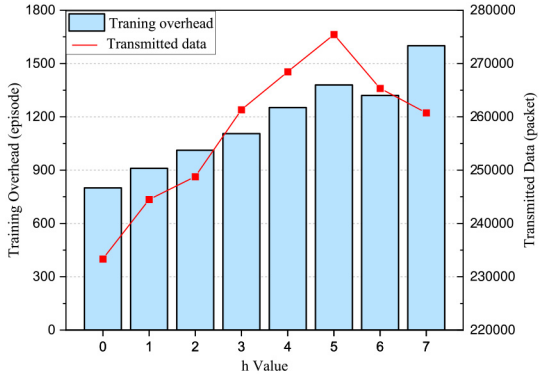


Fig. 6. The effect of h on transmission efficiency and training overhead.

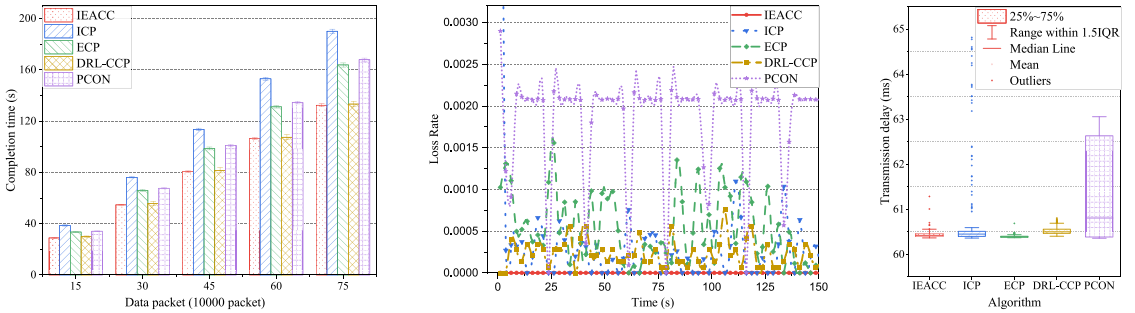
A. Experiment Setting

Before showing the experimental results, we first introduce the basic environment settings and some selection references of important parameters.

1) Basic Environment: To implement the DRL algorithm in ndnSIM, we extend the latest AI interface ns3-AI [43] under ns3 to ndnSIM. Specifically, we add an interaction interface between ndnSIM and reinforcement learning algorithms, which separates computational processes related to the DRL algorithms from data transmission in the pipeline. Thus, the computational complexity of DRL algorithms has very little impact on network communications. Besides, the DNNs included in the framework (i.e., the actor and critic networks) are implemented using PyTorch [44]. Moreover, to show the effect of our scheme, we consider two different topologies, as shown in Fig. 5. The linear scenario has a single interface at the edge router, while in the complex scenario, the edge has three interfaces connected to five content producers. In both scenarios, C1 denotes the consumer that send interests according to congestion algorithms, CP1 to CP5 denote content producers storing different content, and R1 to R8 are routers. BK is a node that sends interest requests randomly to manufacture background flows. We utilize the default routing strategy in ndnSIM and run every testing twenty times to show the results. Moreover, we combine offline pre-training with online selective training and updating to ensure efficiency. In the model training, we constantly change transmission delay, bandwidth, and requested data distribution to simulate various network environments. Generally, the IEACC agent runs for over 1000 episodes containing 250000 transition samples, where the training time varies from ten to twenty hours. When applying the model for congestion control, its computation time is acceptable, about 0.5ms.

2) Parameter Selection: Before the actual tests, we made a performance analysis to select the optimal parameters suitable for the environment. We mainly provide analysis for parameters h in the scheme. It denotes the top h "Later Prefix" introduced in the IRA model.

As shown in Fig. 6, we provide the training overhead and transmission efficiency for eight different values of parameter h. Specifically, we train models for different h values by maintaining other environment settings the same in all scenarios. We record the number of episodes required before the model converges to present the training overhead. Then, we show the transmission efficiency of models by testing the number of data packets transmitted within 50s in the topology shown in Fig. 5(a). From Fig. 6, we can see that the training overhead shows an upward trend as h increases, except for a slight decrease when h is 6. The amount of transmitted data shows a trend of increase then falling, and the transmission efficiency reaches the highest when h is 5. Thus, the parameter h is set



(a) Performance in data completion time. (b) Performance in transmission loss rate. (c) Performance in average transmission delay.

Fig. 7. Transmission performance of different algorithms in stable bandwidth environment.

to 5 to maximize the transmission efficiency in subsequent experiments.

B. Linear Scenario

We first show the efficiency of algorithm in the linear scenario as shown in Fig. 5(a), where we consider stable and randomly varying bandwidth and delay situations.

1) Stable Scenario: In this scenario, we set the bottleneck bandwidth from Edge router to Router to be 50Mbps, and the other paths are all set to 100Mbps. The delay between any two adjacent routers is 10ms.

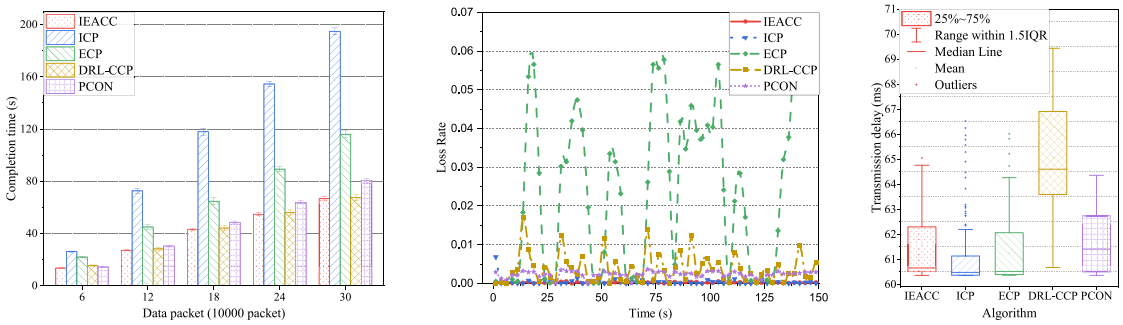
In Fig. 7(a), we first introduce the performance of algorithms in data completion time where the amount of data transmitted ranges from 150000 packets to 750000 packets. We can see that IEACC and DRL-CCP both have high data transmission rates and complete data transmission faster than ICP, ECP, and PCON. Specifically, when the data requirement is 750000 packets, it takes IEACC and DRL-CCP for about 132s to complete the transfer, while ECP and ICP cost 163.8s and 190s, respectively. PCON, with explicit congestion mark, takes 167.804s to complete transfer, which is comparable to ECP and much better than ICP. Thus, compared with the ICP algorithm, IEACC improves the transmission rate by about 32.1%, which is a remarkable improvement. In Fig. 7(b), we show the packet loss of each algorithm during transmission, where the number of packet losses of each algorithm within 150s is relatively low in the stable environment. Among them, IEACC has the lowest packet loss. ICP suffers higher loss packets at the beginning of transmission because of the slow start-up. ECP that adjusts transmission rate when the link queue capacity is occupied for 100% suffers the highest loss. The packet loss rate of PCON that uses interval congestion marking and BIC algorithm for fast recovery shows a regular fluctuations. DRL-CCP with a window larger than the actual requirement because of slower RTT feedbacks also has slightly higher packet loss.

In Fig. 7(c), we show the average transmission delay of each algorithm in a box diagram style, where the five lines from bottom to top represent the minimum, lower quartile, median, upper quartile, and maximum respectively, and small black diamonds are outliers. We can observe that although the data transmission rate is significantly improved, IEACC still has a lower transmission delay. It is because our model adopts the data queue length explicitly transmitted by the intermediate router to convey the degree of congestion. It can effectively and timely control the transmission rate, thereby reducing the data packet queue length to avoid congestion, so that it has a lower transmission delay. Due to the aggressive fast recovery strategy, PCON is unable to empty its queue, so its average transmission delay is the largest.

2) Varying Scenario: In reality, the effective transmission bandwidth and delay of a flow changes with the network traffic traveling through the path. Therefore, we next show the efficiency of algorithms under flexible bandwidth changes and varying transmission delay scenarios to demonstrate the performance of IEACC.

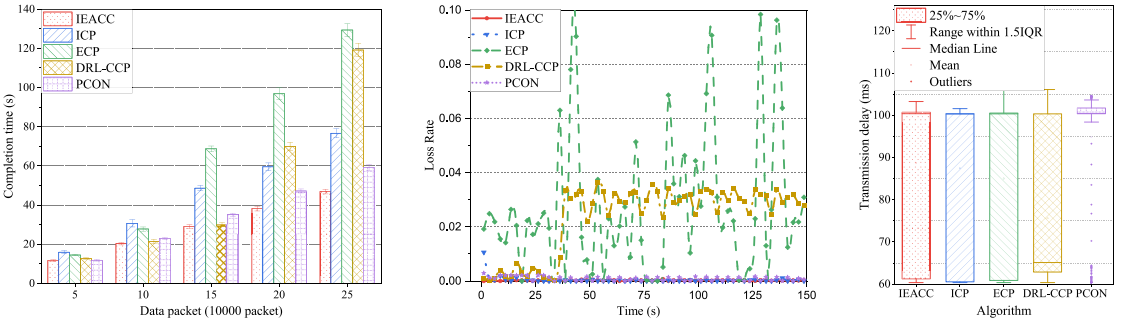
Varying bandwidth: We randomly change the bottleneck bandwidth in the testing to show the performance of IEACC in real network. More specifically, we select twenty timing points in 200s, and then randomly change the bottleneck bandwidth from 20Mbps to 50Mbps.

In Fig. 8, we show the corresponding results. We can see that IEACC, DRL-CCP, and PCON perform better than ICP and ECP. When 300000 packets are transmitted, the completion time of IEACC is 66.63s, and that of DRL-CCP and PCON are 67.55s and 80.76s. But ICP and ECP take 194.83s and 115.99s, respectively. Therefore, compared with ICP, IEACC can reduce the completion time by about 65.8%, which is remarkable. Compared to PCON, IEACC can also reduce transmission time by 17.5%. As for the average delay, algorithms present upward trends compared with that of the stable situation. Among them, DRL-CCP suffers the largest transmission delay with an average value of 64.8ms compared with 60.6ms in IEACC. ICP has a minimal delay because it constantly reduces the window as bandwidth changes randomly. The average transmission delay of PCON is 61.5ms, which is lower than DRL-CCP and higher than other algorithms. The packet loss rate increases distinctly compared with the stable situation, and algorithms also show significant differences. Specifically, the maximal packet loss rate increases from 0.2% to 5.9%, where ECP and DRL-CCP have relatively larger increases. Because ECP utilizes a predefined interest PIT length threshold to detect congestion, it is too stiff to control transmission rate reasonably in a dynamic scenario and thus results in many packet losses. PCON that marks data



(a) Performance in data completion time. (b) Performance in transsmission loss rate. (c) Performance in average transmission delay.

Fig. 8. Transmission performance of different algorithms in dynamic bandwidth environment.



(a) Performance in data completion time. (b) Performance in transmission loss rate. (c) Performance in average transmission delay.

Fig. 9. Transmission performance of different algorithms in varying delay environment.

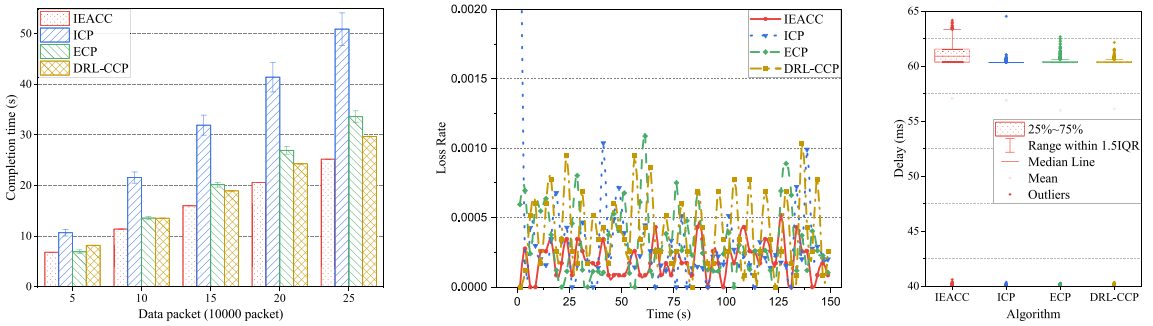
packets according to queue delay performs well in dynamic bandwidth environment, which has a low packet loss rate.

Varying delay: In this scenario, we vary the transmission delay stepwise to show the performance of algorithms. Specifically, we start to change transmission delay after running the algorithm for 30s. Then, we increase the delay at the transmission bottleneck by 1ms every 0.5s until it changes from original 10ms to 30ms, and then remains unchanged.

In Fig. 9, we present the corresponding results. We can see that IEACC and PCON perform very well, which are far better than DRL-CCP when transmission time exceeds 30s. Specifically, when 250000 packets are transmitted, the completion time of IEACC and PCON are 46.87s and 59.24s, while DRL-CCP, ICP, and ECP take 119.19s, 76.68s, and 129,36s, respectively. IEACC utilizes data packet queue length for congestion detection and explicitly conveys the accurate congestion information to DRL model for rate adjustment. However, DRL-CCP utilizes RTT as congestion signal, and it thus suffers misadjustment of congestion and performs terrible in the delay varying scenario. PCON detects congestion by observing the queuing delay of data packets at intermediate routers, which can still make accurate congestion judgments in scenarios where the delay changes, so it performs well. Packet loss rate is shown in Fig. 9(b). We can observe that before starting the delay variation, all the algorithms has a low packet loss rate. After 30 seconds, when transmission delay changes with time, the packet loss rate of DRL-CCP and ECP increases significantly, while IEACC and PCON still perform well for accurate congestion detection. Specifically, compared with performance in a stable network, the maximum packet loss rate in the varying transmission delay scenario increases from 0.2% to 10.2%. The packet loss rate of different algorithms also varies significantly, and the proposed IEACC still performs the best.

There are small differences in transmission delay performance of these algorithms, where they all suffer a wide variation because of varying delay. Compared with other algorithms that empty data queue due to misjudgment, IEACC and PCON have a slightly higher average latency.

To sum up, in the simple linear topology, IEACC performs superb transmission rate control through the DDPG model in both stable and dynamic situations. Besides, it also presents acceptable transmission delay and remarkable transmission reliability with a lower packet loss rate. Compared with IEACC, DRL-CCP presents terrible data transmission rate, suffers from high transmission delay and non-negligible packet loss in the delay varying scenario. Furthermore, compared with methods using DRL, traditional schemes such as ICP and ECP suffer excessive window reduction for using a limited predefined window adjustment method. They usually have poor performance in dynamic bandwidth and varying delay scenarios. However, PCON perform prominently in dynamic networks because of explicit congestion detection methods. However, as a heuristic algorithm that can only implement discrete window adjustment strategies, it still has gaps when compared with IEACC.



(a) Data completion time in complex scenario. (b) Transmission loss rate in complex scenario. (c) Delay in complex scenario.

Fig. 10. Transmission performance of different algorithms in complex scenario.

C. Complex Scenario

In this part, we present the performance of our algorithm in a more complex "net" scenario as shown in Fig. 5(b), where the Edge router has three interfaces, namely Interface0 to Interface2 from top to bottom, connecting to five content produces. The content producers CP2 and CP4 can respond to a part of requests without going to CP3. For IEACC, we maintain a single window at consumer for data requests to multiple produces and implement a DDPG model at edge for three interfaces at the edge router. For other algorithms, we maintain a separate window for each producer, that is, C1 independently maintains three windows for requests to CP1,CP5 and that to CP2/CP3/CP4. Therefore, IEACC will perform more significantly if other algorithms maintain a single window for all content producers as IEACC. As we already showed the great performance of IEACC in the dynamic environment above, we only consider a stable situation in this scenario. Besides, in this part, we present the performance of each algorithm when the response ratio of different CPs changes. Specifically, we vary the content stored in CP2 and CP4 to change the data reply proportion from different CPs.

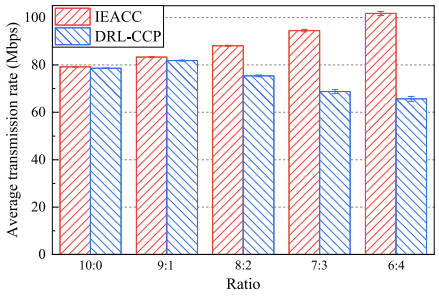
In Fig. 10, we show the performance of each algorithm through completion time, packet loss, and average transmission delay when data respond ratio from CP3 to CP2/CP4 is '8:2'. From Fig. 10(a), we can observe that IEACC utilizes smaller time than the other algorithms including DRL-CCP, which performs as well as IEACC in the linear environment. Taking the data transmission of 150000 packets as an example, the completion time of IEACC and DRL-CCP is 15.9s and 18.94s, respectively, and that of ECP and ICP is 20.15s and 31.93s, respectively. Specifically, compared with DRL-CCP and ICP, IEACC can reduce the completion time by 19.2% and 100.82%. IEACC has a higher data transmission rate mainly because it can obtain more accurate congestion information and adjust the interest rates accordingly to avoid congestion and also pursue higher network utility. However, other algorithms treat data from different congestion levels the same, and thus causes a lower data transmission rate. It needs to note that the performance improvement of IEACC will be more significant if other algorithms maintain a single window at the consumer.

In Fig. 10(b), we also record the packet loss rate in transmission and provide a comparison among different algorithms to verify the transmission reliability of IEACC. We can observe that ICP has a high packet loss at the beginning of simulation because of bandwidth probe at slow-start phase. ECP and DRL-CCP detect congestion by PIT interest length and RTTs, which are easy to make erroneous judgments in a complex scenario that has different types of data. Thus, they have a high loss rate because of error congestion estimation for not distinguishing data with different congestion degrees.

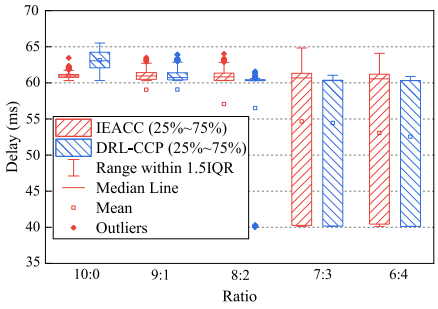
In Fig. 10(c), we compare the average transmission delay of these algorithms, as there are three consumers in other algorithm, we only present the results of Interface1. Among them, IEACC has a relatively higher value, which is 60.9ms, while the others are about 60.03ms. Thus, IEACC increases the transmission delay by about 0.87ms. Indeed, IEACC suffers a little higher delay when facing two different types of data. However, this transmission cost is negligible when it compares to the throughput improvement. We can also observe that all of these algorithms have outliers at 40ms, which is caused by content from intermediate routers that can respond to requests before reaching content producers. Thus, it also proves that NDN with the capability of storing content at intermediate routers has a great advantage in reducing transmission delay compared with TCP/IP networks.

From above analysis, we observe that IEACC that classifies data with their congestion degree and treats them differently can better adapt to the NDN scenarios with multi-source characteristics. In the following, we test different ratio of data distribution to further show the superiority of IEACC. As is shown in Fig. 11, we vary the interest requests to CP3 and CP2/CP4 from '10:0' to '6:4' and show the average transmission rate and delay of IEACC and DRL-CCP, respectively.

From Fig. 11(a) we can observe that the data transmission rate of IEACC increases as the ratio responded at the intermediate router grows. Specifically, when all the data are from the content repository, the transmission rate of IEACC is 79.21Mbps, which increases to 94.41Mbps when the ratio is '7:3'. The rate grows to 101.74Mbps when the ratio is '6:4', which improves the transmission rate by 28.44% compared with the baseline (10:0). Besides, it has a transmission rate improvement of up to 54.94% compared with DRL-CCP when the ratio is '6:4'. This improvement is mainly because of fine congestion detection and control, where it divides all of the



(a) Average transmission rate of IEACCand DRL-CCP.



(b) Average transmission delay of IEACC and DRL-CCP.

Fig. 11. Transmission performance with different data distribution ratio.

coming data into categories and treats them differently. Thus, when there is no congestion, IEACC may grow rate more aggressively by considering the proportion of data that does not need to go through the bottleneck to obtain data. When the model detects congestion, it decreases rate more reasonably rather than simply reducing rate by half and avoids excessive traffic reduction of the non-congested path.

Compared with IEACC with data classification, DRL-CCP performs worse when the ratio increases. The transmission rate of DRL-CCP increase from 78.64Mbps at baseline to 81.85Mbps when the ratio is '9:1'. However, as the ratio further increases, the data transmission rate decreases. More specifically, its throughput decreases to 68.74Mbps and 65.67Mbps at the ratio of '7:3' and '6:4', respectively. That is, DRL-CCP decreases the transmission rate by about 16.49% where data responded at intermediate routers occupies sixty percent of all data. The main reason for a transmission rate decrease is that DRL-CCP treats each data the same and utilizes RTT for congestion signal leading to misjudgement on congestion. When a small part of data from intermediate routers with a smaller RTT, DRL-CCP will increase its window, as the smaller RTT is a signal of good path condition. However, when a large proportion of data comes from intermediate routers, DRL-CCP will compute a reasonable window suitable to the smaller RTT for congestion control. Thus, when data from content producers with a higher RTT, it will decrease the window despite the fact that there might be no congestion.

In Fig. 11(b), we show average delay in this scenario. In the baseline where the ratio is '10:0', IEACC has a lower transmission delay than DRL-CCP, which reduces by 3.5s. When

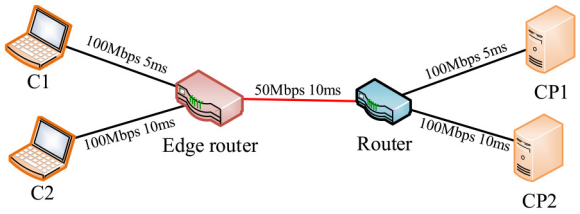


Fig. 12. The topology of fairness testing.

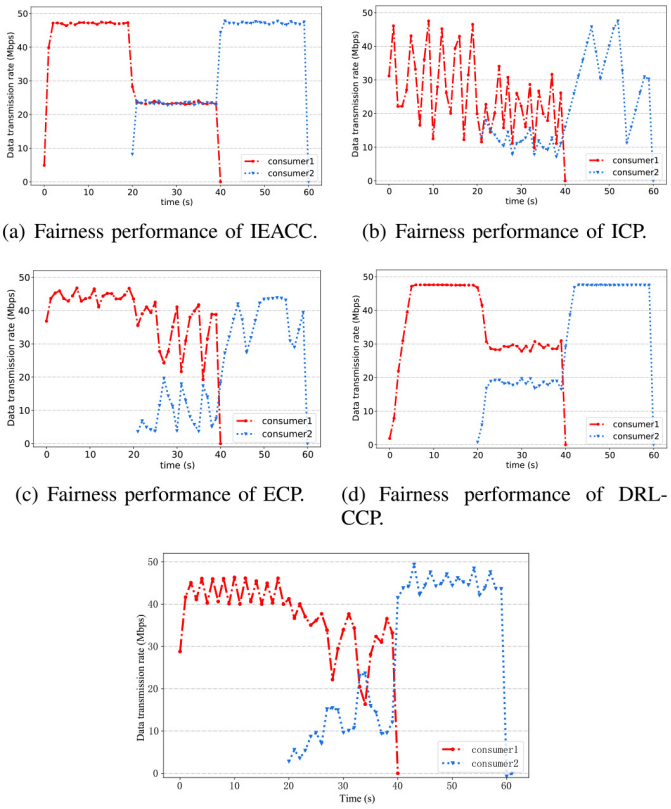
the ratio is less than '8:2', the transmission delay of IEACC increases with the proportion of data answered at intermediate routers, which is slightly larger than DRL-CCP in the scenario of '8:2'. When the ratio continues to rise, the packets from intermediate routers that have less transmission delay also grows, so both IEACC and DRL-CCP have lower delays. On average, DRL-CCP has a slightly smaller transmission delay than IEACC when the ratio increases. However, it is not because of the advantages of great congestion control. It is mainly because the unreasonable window adjustment leads to fewer packets transmitted along the path, which decreases the transmission queue, resulting in decreased transmission delay.

We can summarize that IEACC treats data differently according to the congestion information monitored by the PCD module, which is suitable for the multi-source characteristic of NDN. It can not only adjust the interest rate to avoid congestion at the bottleneck but also maximize network utility by considering that some interests may be responded to at intermediate routers without going through congestion paths. Compared with the DRL-CCP scheme that also uses DRL, IEACC not only improves data transmission rate remarkably but also maintains good transmission reliability with a smaller packet loss. As for the performance of transmission delay, there is not much difference.

D. Fairness

In this part, we show the performance of IEACC in fairness. The topology we used is as shown in Fig. 12, where we connect two consumers to the edge router. We dynamically control consumers to stat or complete transmission to show the performance of FRA in dynamic environments. Specifically, consumer C1 transmits at the beginning of the transmission simulation, and consumer C2 joins the transmission at 20s later. They end their transmission at 40s and 60s, respectively. In order to show the difference between popular content and unpopular content, consumer C1 requests popular content that is responded at the CP1, while consumer C2 requests unpopular content responded at the CP2 . The transmission delay of popular and unpopular content is 40ms and 60ms, respectively.

In Fig. 13, we show packet transmission rate of each consumer. IEACC performs the best among the five considered algorithms, while the other four algorithms present various degrees of unfairness. ICP, ECP, and PCON have similar performance. When the new consumer joins, they suffer great transmission fluctuation, and the transmission rate of consumer C1 is significantly higher than that of consumer C2 because the window of consumer C1 with shorter transmission delay grows



(a) Fairness performance of IEACC.

(b) Fairness performance of ICP.

(c) Fairness performance of ECP.

(d) Fairness performance of DRL-CCP.

(e) Fairness performance of PCON.

Fig. 13. Fairness performance of different algorithms.

faster. DRL-CCP changes transmission rate smoothly when a new consumer joins, which can maximize network utilization as IEACC. However, its popular content obtains a higher transmission rate, which is unfair. IEACC can achieve fairness perfectly, even when consumers have different transmission delays by adjusting consumers' transmission rate in time. When a consumer completes the transmission, the remaining consumer can increase its rate to maximize the network utilization. Besides, consumers can reach the rate allocated by the edge router in a short time because they double their rate at the beginning of a transmission.

VI. Conclusion

In this paper, we proposed a congestion control scheme, named IEACC, for the NDN network. It consists of PCD, IRA, and FRA to ensure accurate congestion detection, obtain a reasonable transmission rate, and maintain fairness, respectively. Specifically, PCD enables intermediate routers to attach congestion status to passing data packets and convey them to consumers. IRA provides reasonable transmission rates through the DRL agent, for which the edge classifies data packets into different congestion degrees by the lightweight clustering algorithm to provide suitable inputs. FRA equitably distributes the resources estimated by the DRL agent to consumers with transmission needs to maintain fairness. Our experiments show that IEACC performs better than traditional algorithms, such as ICP and ECP, in terms of transmission rate in linear and complex topologies. At the same time, IEACC considering the multi-source characteristic of NDN has a transmission rate improvement of up to 34.84% compared with DRL-CCP.

References

[1] "Mobile Data Traffic Outlook." [Online]. Available: https://www.ericsson.com/en/mobility-report/dataforecasts/mobile-traffic-forecast/ (Accessed: Jul. 2022).

[2] G. Xylomenos *et al.*, "A survey of information-centric networking research," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 2, pp. 1024–1049, 2nd Quart., 2014.

[3] L. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 3, pp. 66–73, Jul. 2014.

[4] D. Posch, B. Rainer, and H. Hellwagner, "SAF: Stochastic adaptive forwarding in named data networking," *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 1089–1102, Apr. 2017.

[5] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proc. ACM Int. Conf. Appl. Technol. Archit. Protocols Comput. Commun. (SIGCOMM)*. 1998, pp. 303–314.

[6] Y. Ren, J. Li, S. Shi, L. Li, G. Wang, and B. Zhang, "Congestion control in named data networking—A survey," *Comput. Commun.*, vol. 86, pp. 1–11, Jul. 2016.

[7] G. Carofiglio, M. Gallo, and L. Muscariello, "ICP: Design and evaluation of an interest control protocol for content-centric networking," in *Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2012, pp. 304–309.

[8] M. Amadeo, A. Molinaro, C. Campolo, M. Sifalakis, and C. Tschudin, "Transport layer design for named data wireless networking," in *Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2014, pp. 464–469.

[9] Y. Ye *et al.*, "PTP: Path-specified transport protocol for concurrent multipath transmission in named data networks," *Comput. Netw.*, vol. 144, pp. 280–296, Oct. 2018.

[10] F. Wu, W. Yang, M. Sun, J. Ren, and F. Lyu, "Multi-path selection and congestion control for NDN: An online learning approach," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 2, pp. 1977–1989, Jun. 2021.

[11] N. Rozhnova and S. Fdida, "An effective hop-by-hop interest shaping mechanism for CCN communications," in *Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2012, pp. 322–327.

[12] Y. Wang, N. Rozhnova, A. Narayanan, D. Oran, and I. Rhee, "An improved hop-by-hop interest shaper for congestion control in named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 55–60, 2013.

[13] G. Carofiglio, M. Gallo, and L. Muscariello, "Joint hop-by-hop and receiver-driven interest control protocol for content-centric networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 491–496, 2012.

[14] S. N. S. Hashemi and A. Bohlooli, "3CP: Coordinated congestion control protocol for named-data networking," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 3, pp. 3918–3932, Sep. 2021.

[15] K. Schneider, C. Yi, B. Zhang, and L. Zhang, "A practical congestion control scheme for named data networking," in *Proc. ACM Conf. Inf.-Centric Netw. (ICN)*, 2016, pp. 21–30.

[16] D. Lan, X. Tan, J. Lv, Y. Jin, and J. Yang, "A deep reinforcement learning based congestion control mechanism for NDN," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2019, pp. 1–7.

[17] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. ACM Int. Conf. Emerg. Netw. Exp. Technol. (CoNEXT)*, 2009, pp. 1–12.

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

[19] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[20] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015, *arXiv1509.02971*.

[21] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2014, pp. 387–395.

[22] C. Chen, C. Wang, T. Qiu, M. Atiquzzaman, and D. O. Wu, "Caching in vehicular named data networking: Architecture, schemes and future directions," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 4, pp. 2378–2407, 4th Quart., 2020.

[23] S. Misra, R. Tourani, F. Natividad, T. Mick, N. E. Majd, and H. Huang, "AccConF: An access control framework for leveraging in-network cached data in the ICN-enabled wireless edge," *IEEE Trans. Dependable Secure Comput.*, vol. 16, no. 1, pp. 5–17, Jan./Feb. 2019.

[24] K. Xue *et al.*, "A secure, efficient, and accountable edge-based access control framework for information centric networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 3, pp. 1220–1233, Jun. 2019.

[25] M. Amadeo, G. Ruggeri, C. Campolo, and A. Molinaro, "IoT services allocation at the edge via named data networking: From optimal bounds to practical design," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 661–674, Jun. 2019.

[26] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. PMLR Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 1928–1937.

[27] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 1587–1596.

[28] S. Salsano, A. Detti, M. Cancellieri, M. Pomposini, and N. Blefari-Melazzi, "Receiver-driven interest control protocol for content-centric networks," in *Proc. ACM SIGCOMM Workshop Inf. Centric Netw. (ICN)*, 2012, pp. 37–42.

[29] Y. Ren, J. Li, S. Shi, L. Li, and G. Wang, "An explicit congestion control algorithm for named data networking," in *Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2016, pp. 294–299.

[30] L. Saino, C. Cocora, and G. Pavlou, "CCTCP: A scalable receiver-driven congestion control protocol for content centric networking," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2013, pp. 3775–3780.

[31] G. Carofiglio, M. Gallo, L. Muscariello, and M. Papali, "Multipath congestion control in content-centric networks," in *Proc. IEEE Int. Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2013, pp. 363–368.

[32] S. Braun, M. Monti, M. Sifalakis, and C. Tschudin, "An empirical study of receiver-based AIMD flow-control strategies for CCN," in *Proc. IEEE Int. Conf. Comput. Commun. Netw. (ICCCN)*, 2013, pp. 1–8.

[33] M. Nikzad, K. Jamshidi, and A. Bohlooli, "A responsibility-based transport control for named data networking," *Future Gener. Comput. Syst.*, vol. 106, pp. 518–533, May 2020.

[34] G. Carofiglio, M. Gallo, and L. Muscariello, "Optimal multipath congestion control and request forwarding in information-centric networks: Protocol design and experimentation," *Comput. Netw.*, vol. 110, pp. 104–117, Dec. 2016.

[35] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 3, pp. 62–67, 2012.

[36] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM 2.0: A new version of the NDN simulator for NS-3," Dept. Comput. Sci., Univ. California, Los Angeles, CA, USA, Rep. NDN-0028, 2015. Accessed: Jul. 2022. [Online]. Available: https://named-data.net/wp-content/uploads/2013/07/ndn-0028-1-ndnsim-v2.pdf

[37] S. Floyd, T. Henderson, and A. Gurtov, "The NewReno modification to TCP's fast recovery algorithm," IETF, RFC 2582, 2021. Accessed: Jul. 2022. [Online]. Available: https://www.ietf.org/rfc/rfc2582.txt

[38] S. Ha, I. Rhee, and L. Xu, "CUBIC: A new TCP-friendly high-speed TCP variant," *ACM Special Interest Group Oper. Syst. Rev.*, vol. 42, no. 5, pp. 64–74, 2008.

[39] Y. G. Iyer, S. Gandham, and S. Venkatesan, "STCP: A generic transport layer protocol for wireless sensor networks," in *Proc. Int. Conf. Comput. Commun. Netw. (ICCCN)*, 2005, pp. 449–454.

[40] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 1999, pp. 1057–1063.

[41] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Comput. Netw. ISDN Syst.*, vol. 17, no. 1, pp. 1–14, 1989.

[42] R. Adams, "Active queue management: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1425–1476, 3rd Quart., 2013.

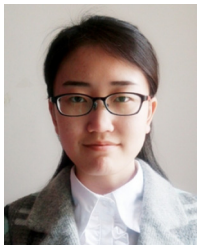
[43] H. Yin *et al.*, "NS3-AI: Fostering artificial intelligence algorithms for networking research," in *Proc. ACM Workshop NS-3 (WNS3)*, 2020, pp. 57–64.

[44] "Pytorch." [Online]. Available: https://pytorch.org/ (Accessed: Jul. 2022).

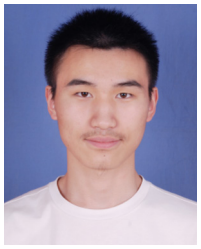
Jiayu Yang (Student Member, IEEE) received the bachelor's degree in information security from the School of Cyber Science and Technology, University of Science and Technology of China in 2019, where she is currently pursuing the Ph.D. degree. Her research interests include future Internet architecture design, transmission optimization, and network security.

Yuxin Chen received the bachelor's degree in information security from the School of the Gifted Young, University of Science and Technology of China (USTC) in July 2021. He is currently a Graduated Student with the School of Cyber Science and Technology, USTC. His research interests include future Internet architecture design, transmission optimization, and network security.

Kaiping Xue (Senior Member, IEEE) received the bachelor's degree from the Department of Information Security, University of Science and Technology of China (USTC) in 2003, and the Ph.D. degree from the Department of Electronic Engineering and Information Science, USTC in 2007. From May 2012 to May 2013, he was a Postdoctoral Researcher with the Department of Electrical and Computer Engineering, University of Florida. He is Currently a Professor with the School of Cyber Science and Technology, USTC. His research interests include next-generation Internet architecture design, transmission optimization, and network security. He serves on the Editorial Board of several journals, including the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT. He has also served as a (Lead) Guest Editor for many reputed journals/magazines, including IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, *IEEE Communications Magazine*, and IEEE NETWORK. He is an IET Fellow.



Jiangping Han received the bachelor's and Ph.D. degrees from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC) in 2016 and 2020, respectively. From November 2019 to October 2021, She was a Visiting Scholar with the School of Computing, Informatics, and Decision Systems Engineering, Arizona State University. She is currently a Post-Doctoral Researcher with the School of Cyber Science and Technology, USTC. Her research interests include future Internet architecture design and transmission optimization.



Jian Li (Member, IEEE) received the bachelor's degree from the Department of Electronics and Information Engineering, Anhui University in 2015, and the Ph.D. degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC) in 2020. From November 2019 to November 2020, he was a Visiting Scholar with the Department of Electronic and Computer Engineering, University of Florida. He is currently a Post-Doctoral Researcher with the School of Cyber Science and Technology, USTC. His research interests include wireless communications, satellite networks, and next-generation Internet.

David S. L. Wei (Life Senior Member, IEEE) received the Ph.D. degree in computer and information science from the University of Pennsylvania in 1991. From May 1993 to August 1997, he was on the faculty of Computer Science and Engineering with the University of Aizu, Japan (as an Associate Professor and then a Professor). He is currently a Professor of Computer and Information Science Department with Fordham University. He has authored and coauthored more than 120 technical papers in various archival journals and conference proceedings. His research interests include cloud computing, big data, IoT, and cognitive radio networks. He was a Guest Editor or a Lead Guest Editor for several special issues in the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE TRANSACTIONS ON CLOUD COMPUTING, and the IEEE TRANSACTIONS ON BIG DATA. He also served as an Associate Editor for IEEE TRANSACTIONS ON CLOUD COMPUTING from 2014 to 2018 and *Journal of Circuits, Systems and Computers* from 2013 to 2018.

Qibin Sun (Fellow, IEEE) received the Ph.D. degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC) in 1997. From 1996 to 2007, he was with the Institute for Infocomm Research, Singapore, where he was responsible for industrial as well as academic research projects in the area of media security and image and video analysis. He was the Head of Delegates of Singapore in ISO/IEC SC29 WG1(JPEG). He worked as a Research Scientist with Columbia University from 2000 to 2001. He is Currently a Professor with the School of Cyber Science and Technology, USTC. He has published more than 120 papers in international journals and conferences. His research interests include multimedia security, network intelligence, and security.

Jun Lu received the bachelor's degree from southeast university in 1985, and the master's degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC) in 1988, where he is Currently a professor. His research interests include theoretical research and system development in the field of integrated electronic information systems. He is an Academician of the Chinese Academy of Engineering.