electronics

MDPI

Article

An Effective Transmission Scheme Based on Early Congestion Detection for Information-Centric Network

Yong Xu [1,2], Hong Ni [1,2] and Xiaoyong Zhu [1,2,*]

[1] National Network New Media Engineering Research Center, Institute of Acoustics, Chinese Academy of Sciences, No. 21, North Fourth Ring Road, Haidian District, Beijing 100190, China; xuy@dsp.ac.cn (Y.X.); nih@dsp.ac.cn (H.N.)

[2] School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, No. 19(A), Yuquan Road, Shijingshan District, Beijing 100049, China

* Correspondence: zhuxy@dsp.ac.cn; Tel.: +86-131-2116-8320

Abstract: As one of the candidates for future network architecture, Information-Centric Networking (ICN) has revolutionized the manner of content retrieval by transforming the communication mode from host-centric to information-centric. Unlike a traditional TCP/IP network, ICN uses a location-independent name to identify content and takes a receiver-driven model to retrieve the content. Moreover, ICN routers not only perform a forwarding function but also act as content providers due to pervasive in-network caching. The network traffic is more complicated and routers are more prone to congestion. These distinguished characteristics pose new challenges to ICN transmission control mechanism. In this paper, we propose an effective transmission scheme by combining the receiver-driven transport protocol and the router-driven congestion detection mechanism. We first outline the process of content retrieval and transmission in an IP-compatible ICN architecture and propose a practical receiver-driven transport protocol. Then, we present an early congestion detection mechanism applied on ICN routers based on an improved Active Queue Management (AQM) algorithm and design a receiver-driven congestion control algorithm. Finally, experiment results show that the proposed transmission scheme can maintain high bandwidth utilization and significantly reduce transmission delay and packet loss rate.

Keywords: ICN; receiver-driven; congestion control; congestion detection; active queue manage

Citation: Xu, Y.; Ni, H.; Zhu, X. An Effective Transmission Scheme Based on Early Congestion Detection for Information-Centric Network. *Electronics* **2021**, *10*, 2205. https://doi.org/10.3390/electronics10182205

Academic Editor: Stavros Shiaeles

Received: 28 July 2021
Accepted: 7 September 2021
Published: 9 September 2021

1. Introduction

The current Internet architecture is established on the TCP/IP protocol family where communication is based on named hosts. However, with the emergence of new applications and services on the Internet and the explosive growth of IP traffic [1], the traditional TCP/IP network is facing a series of problems such as scalability, mobility and security [2]. The existing end-to-end communication model has been unable to meet the needs of massive content distribution. The usage patterns of the Internet have greatly changed. Compared with the physical or logical location of data, users are more concerned about the data themselves. Although Content Delivery Networks (CDN) and Peer-to-Peer (P2P) [3] have been proposed to solve some of the problems by using the concept of information-centric content retrieval, their performance is still limited due to their deployment at the application layer.

In this context, Information-Centric Networking (ICN), as one of the candidates for future network architecture, has been proposed to tackle these challenges. ICN transforms the communication mode from host-centric to information-centric. Following this principle, ICN decouples content addressing from routing by separating content identifiers (ID) and network address (NA), and users can address and locate the content through the ID without connecting to a specific server. The named content can be widely cached on the intermediate routers, which makes the routers become content providers. The users' requests can be

served by the intermediate routers, which reduces the delay of downloading content and the load of the servers, thereby realizing efficient content distribution.

Some ICN approaches deploy clean-slate ICN, such as Content Centric Networking (CCN) [2], Named Data Networking (NDN) [4,5] and Publish Subscribe Internet Technology (PURSUIT) [6]. However, the existing Internet architecture is entrenched. The deployment of a clean-state ICN requires a major upgrade of the entire network infrastructure [7], which is undoubtedly costly. To reduce the cost of deployment and smoothly evolve, some researchers believe that the ICN approach should be compatible with existing network to utilize IP facilities and routing schemes [8], such as Data-Oriented Network Architecture (DONA) [9], MobilityFirst [10], Network of Information (NetInf) [11], and On-Site, Elastic, Autonomous Network (SEANet) [12]. Readers should note that the transmission scheme we propose in this paper is based on the latter ICN architecture.

In ICN, data are usually regarded as a Named Data Chunk (NDC), which is an addressable data unit bound with a globally unique name. The NDC can be widely cached on ICN routers and users adopt pull-based methods to obtain NDC from multiple content sources. The ICN routers not only simply perform forwarding functions, but also are content providers, which makes the ICN routers have two types of traffic, including external forwarding traffic and internal cache service traffic. The traffic competition between the two types of traffic can easily lead to congestion [13]. Therefore, the transmission in ICN has the characteristics of receiver-driven, multi-source and multi-path, and the traffic behavior is more complicated.

Content distribution is one of the key issues that need to be researched in ICN, and a reasonable transmission scheme can improve the efficiency of content distribution. In ICN, most studies on transmission control are focused on the receiver side [14]. Some researchers proposed a method similar to TCP end-to-end congestion control mechanism, that is, the receiver adjusts the request rate by maintaining one or more congestion windows or timers and detects congestion through round-trip time (RTT) or packet loss [15–19]. However, the receiver needs to maintain much state information for multiple sources or paths. Thus, their cost is high [20]. Moreover, some researchers proposed to monitor the queue size or available bandwidth of intermediate routers to detect congestion [20–25] such as Active Queue Management (AQM) algorithms in TCP/IP network, and then notify the receiver by explicitly marking data packets or negative acknowledgement (NACK). This method is similar to the explicit congestion notification (ECN) in the IP network. However, the routers do not directly drop packets, so it is difficult to quickly recover from a heavy congestion state [13]. In addition, they mechanically apply the AQM algorithm of IP network to ICN network, ignoring the difference in traffic between ICN and IP network. Therefore, in this paper, we fully consider the characteristics of the ICN transmission mode and traffic behavior and propose an effective transmission scheme based on early congestion detection. The proposed transmission scheme includes a receiver-driven transport protocol and a router-driven congestion detection mechanism. The main contributions of our work are as follows:

To ensure the reliable and efficient transmission service for NDC, we design and implement a receiver-driven transport protocol in an IP-compatible ICN architecture. The basic functions of the transport protocol, including the transmission process, the protocol stack, the retransmission mechanism, and the congestion control algorithm, are described in detail.

We further propose an early congestion detection mechanism applied on ICN routers based on an improved AQM algorithm. According to the queuing delay of packets, we divide congestion into two phases: early congestion and permanent congestion. The router takes the marking and dropping operations, respectively in these two phases and the receiver takes the gentle and drastic congestion control methods, respectively according to the marking signal and loss signal.

We conduct extensive experiments to evaluate the performance of our transmission scheme. The experimental results show that the proposed transmission scheme has good

performance in terms of bandwidth utilization, fairness, buffer occupancy, and packet loss rate. In addition, we evaluate our proposal in ICN scenario and the results show that the proposed transmission scheme has the shortest flow completion time and the lowest average content download delay compared with other schemes.

The remainder of the paper is organized as follows. In Section 2, we review the related work of the congestion control mechanism and the congestion detection mechanism. The complete receiver-driven transport protocol is introduced in Section 3. Section 4 presents the early congestion detection mechanism based on an improved AQM algorithm. Then, the experimental results and analysis are shown in Section 5. Finally, we make a conclusion of this paper in Section 6.

2. Related Work

Since the transmission mode in ICN is quite different from the existing TCP/IP network, it is necessary to redesign the transmission scheme for ICN to improve the efficiency of content distribution. The transmission scheme we propose includes two key issues: congestion control mechanism at receiver side and congestion detection mechanism at intermediate routers. In this Section, we conduct a comprehensive study of the current literature on them.

2.1. Congestion Control Mechanism

As ICN shifts the content retrieval model from host-centric to information-centric, the logic of congestion control mechanism for the ICN is usually placed on the receiver side.

Information-Centric Transport Protocol (ICTP) [15] is one of the earliest transport protocols for ICN, which adopts TCP-like algorithm to adjust congestion window through additive increase multiplicative decrease (AIMD) algorithm. The authors also discussed the problem of the proper chunk size to realize a balance between safety overhead and flow control efficiency. In [16], the authors proposed the Interest Control Protocol (ICP) as the receiver-driven transport protocol for CCN. ICP uses a window-based interest control mechanism and detects congestion according to whether the retransmission timeout (RTO) timer expires or not. The authors analyzed the fact that ICP can achieve optimal bandwidth sharing and efficient bandwidth utilization. However, the transmission mode of ICN is multi-source, and ICP and ICTP do not perform well in RTT estimation. Then, Remote Adaptive Active Queue Management (RAAQM) [17] was proposed to enhance the robustness of RTT estimation. RAAQM monitors RTT on each path and realizes efficient multipath congestion control. Unlike RAAQM, ConTug [18] and Content Centric TCP (CCTCP) [19] maintain an independent timer for each content source to solve the problem of inaccurate RTT estimation in CCN. ConTug assumes that the receiver knows the location of the content source, and the location remains the same during transmission, which violates the basic concept of ICN. CCTCP adopts a method to predict the location of the content source and calculates the RTO based on the predicted RTT between each content source and receiver. However, these methods require the receiver to maintain much state information, increasing the complexity of congestion control. NetInf TP [26] is designed as a receiver-driven transmission protocol in NetInf, which uses cyclic retransmission and TCP-like congestion control to provide reliable data transmission. The simulation results showed that NetInf TP can achieve high bandwidth utilization and fairness with TCP flows or other concurrent NetInf TP flows. In [27], the authors proposed a delay-based congestion control protocol (DCP), which measures the delay of the forwarding path from content sources to receiver to detect and control congestion at the receiver side. Likewise in [28], the authors designed and implemented a delay-based transmission mechanism in the wireless scenario, and applies the Bottleneck Bandwidth and Round-trip-time (BBR) algorithm in the receiver-driven context for the first time. Experimental results showed that the proposed method has a high bandwidth utilization and low propagation delay. In [29], the authors proposed a two-level congestion control mechanism, which first selects the best replica and then performs rate control at the receiver. However, the above congestion

control methods are based on the receiver side, and relying on only RTT variation or packet loss to detect congestion is inaccurate and untimely.

2.2. Congestion Detection Mechanism

The internal traffic of ICN is dynamic and complex due to ubiquitous in-network caching. Only relying on the end system to detect congestion is not accurate [13], so the congestion detection mechanism based on intermediate nodes is necessary.

The AQM algorithm can be used for congestion detection, and it has been extensively studied in the past 20 years. Drop Tail is a commonly used Passive Queue Management (PQM) algorithm. Generally, when the queue length reaches the limit threshold, the enqueue packets will be discarded until the queue length converges within the threshold. The Random Early Detection (RED) [30] is the first AQM algorithm which sets minimum and maximum thresholds for queue length on routers. When the average queue length is between the minimum and maximum threshold, the enqueue packets are dropped with a variable probability. The performance of RED depends on the setting of parameters and the fairness is not guaranteed [31], leading to it having poor adaptability in the variable network environment. In this context, many improved versions have been derived, such as Gentle-RED [32], ARED [31]. Unlike RED, BLUE [33] detects network congestion through packet loss event and link idle event, and sets a separate probability to mark or drop enqueue packets based on the detection results. In addition, some researchers used control theory for queue management. Misra et al. [34] established a nonlinear fluid flow model of TCP/AQM system by using stochastic differential equations and analyzed the dynamic characteristics of queue size in routers. In [35], the proportional integral (PI) controller is proposed to maintain the queue stability, but it has a certain hysteresis in queue size changes. Moreover, the queue-based AQM schemes have been studied from the perspective of chaos theory that also solve congestion control problems by using bifurcation analysis [36–38]. In [36], the authors conduct bifurcation analysis on the nonlinear time-delay model for congestion control process of TCP/AQM network based on control theory and bifurcation theory. The local dynamics around the equilibrium of a class of proportional feedback TCP/AQM network is studied and the influence of communication delay on network stability is discussed in the literature [37]. In [38], the bifurcation and chaotic behavior of the fluid model of the TCP/AQM network congestion control system are studied, and an impulse control method is proposed. In [39], the authors proposed an innovative queue management method, Controlled Delay (CoDel), according to the queuing delay instead of the queue length. CoDel simply maintains the target and the interval parameters and allows the sojourn time of packets to be higher than the target for one interval. However, the drastic dropping operation will cause a high loss rate in the dropping state. Although some researchers attempted to improve CoDel by dropping packets in advance [40], the operation will affect the performance of the congestion algorithm at end side. Readers should note that the proposed early congestion detection mechanism in this paper is based on Codel algorithm.

In ICN, there have been some attempts to apply the idea of an AQM algorithm to detect congestion. Some researchers propose hop-by-hop congestion control mechanism which requires the intermediate routers to participate in the rate control process. HoBHIS [20] is the first hop-by-hop congestion control method which calculates the interest forwarding rate based on the buffer occupancy of routers. In [21], the authors proposed that the routers adjust the interest forwarding rate in the upstream direction according to the available bandwidth in the downstream direction. Although the hop-by-hop method can adjust the forwarding rate according to the real-time congestion state, it also greatly increases the calculation pressure of the router. In addition, some researchers use the idea of AQM algorithm to detect congestion and assist the receiver to adjust the request rate through ECN. In [22], the congestion degree of the forwarding port is simply marked with three colors of red, yellow, and green by monitoring the queue length, indicating that the port is not congested, may be congested, and congested, respectively. When congestion occurs,

the congested node will generate a NACK signal and forward it to the downstream nodes. NACK mechanism enables routers to respond to the changes in link state in a timely manner. Explicit Congestion control Protocol (ECP) [23] is proposed to measure the congestion level in NDN by monitoring the average interest queue size of each interface. According to the average interest queue size, ECP classifies the network congestion state into three levels (namely Free Link, Busy Link and Congested Link), and guide the receiver to shape its interest sending rate by ECN. In [24], the authors proposed to use random early marking (REM), which marks the data packets with a variable probability according to the queue size, to indicate the degree of congestion. In [25], a practical congestion control scheme, PCON, is proposed based on the CoDel algorithm to detect congestion. PCON signals congestion by explicitly marking data packets, so that downstream routers can divert traffic to other paths. However, these congestion detection methods mentioned above mechanically apply the traditional AQM algorithm of TCP/IP network to ICN network, ignoring the difference in traffic characteristics between the ICN and the TCP/IP network.

Inspired by the above research, the design of the ICN transmission scheme should fully consider the characteristics of the transmission mode and traffic behavior. In this context, we provide an effective transmission scheme based on early congestion detection. By detecting early congestion and permanent congestion at intermediate nodes, the receiver can make the correct response according to the network status.

3. Proposed Transport Protocol

In this section, we first outline the transmission process based on an IP-compatible ICN network. Then, we detail a receiver-driven transport protocol, including the design of protocol, retransmission mechanism and congestion control mechanism.

3.1. Overview of Transmission Process

In order to reduce the cost of deployment and smoothly evolve, the new ICN architecture should be incrementally deployed on existing networks [7]. The ID and NA are separated in ICN and the name resolution system (NRS) is an important network component that stores the mapping between them [41]. We use a flat ID to name content or device and the ID has the characteristics of self-verification. Considering that the existing TCP/IP network is entrenched, using an IP address as a NA is a reasonable option [29]. Therefore, the users' requests can be addressed through looking the corresponding IP on the NRS by content ID, and then forwarded to the content source via IP routing function. Readers should note that late-binding technology [42] is applied in our scenario, that is, the intermediate node can make a name resolution request to the NRS and change the IP of the packet during forwarding. Considering the large overhead that small NDC impose on transmission and caching [43], the NDC size is set to several MBs. Additionally, we consider a hybrid network of IP and ICN and the entire network consists of ICN routers and IP routers. In [44], the authors designed a cache-enabled router in this kind of ICN architecture by adding a content store (CS) module with a complete ICN protocol stack on routers, so the routers can handle both IP flows and ICN flows.

Figure 1 shows the overview of the transmission process based on an IP-compatible ICN network. The user builds the request packets based on the ID of the NDC to be requested and sends it to the edge router (R1) (Step 1). For security reasons, the terminal cannot obtain the NA of the nodes inside the network, so the name resolution process is performed at the edge nodes. Specifically, R1 sends a name resolution request to the NRS based on the ID of NDC to obtain the list of NAes (NA2, NA4 and NA6) of cache routers or servers. Then, R1 selects an optimal NA from the list through the multi-source selection strategy. Typically, the nearest node (here is R2) is selected (Step 2). Since IP is regarded as NA in our scenario, R1 fills the selected NA into the destination network address field of the request packet and sends it to the corresponding cache router (R2) through the IP routing function (Step 3). While receiving the users' request, R2 segments the NDC according to the Maximum Transmission Unit (MTU). These segments are encapsulated into ICN

packets and then sent back to the receiver (Step 4). During the transmission process, the routers (for example, R1) on or off the path actively decide whether to cache the NDC or not according to the caching strategy (Step 5). The receiver performs congestion control and retransmission during the whole transmission, and finally assembles the received ICN packets into a complete NDC (Step 6).

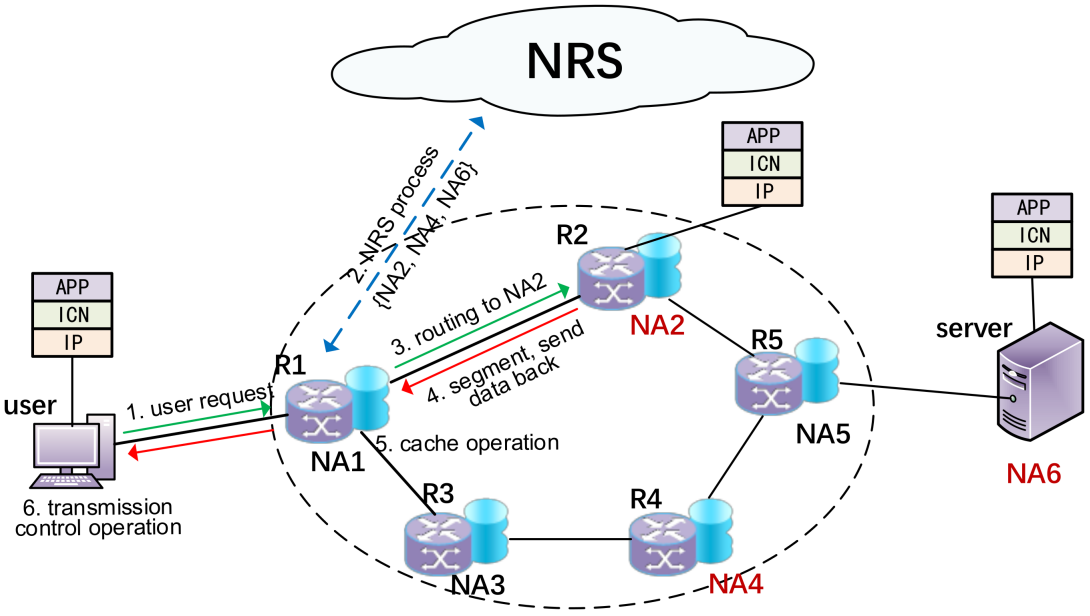

Figure 1. The overview of the transmission process in ICN.

3.2. Protocol Design

3.2.1. Protocol Description

Figure 2 shows a practical ICN protocol stack that is compatible with IP network. The functionality of the network layer is extended by placing an ID layer on top of the IP layer. Considering that the existing TCP/IP network intermediate nodes are basically two-layer or three-layer structure, the ICN packets can be routed and forwarded normally on the existing network. Therefore, our proposal is theoretically compatible with most network architectures which are built on TCP/IP. There is a transport layer on top of the ID layer, and we design a receiver-driven transport protocol which can provide reliable and efficient transmission service for NDC.

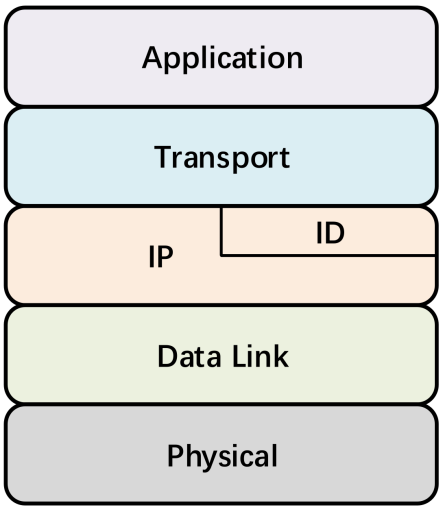

Figure 2. The protocol stack of IP-compatible ICN network.

The brief design of the packet format of an ICN header is shown in Figure 3. There is a "Protocol" field in the IP header, which is used to indicate what protocol the IP payload belongs to. For example, TCP corresponds to a "Protocol" field value of 6, and UDP corresponds to a "Protocol" field value of 17. We set the value of the "Protocol" field to a 0×99 to indicate that the next header represents the ID header. The ID layer contains two important fields, "Source ID" and "Destination ID", which represent the ID of the receiver's device and the ID of NDC to be retrieved, respectively. The NDC transmission is labelled by these two location-independent IDs. The pair of IDs remains the same during the whole NDC transmission, even though the NAes have changed (multisource selection mechanism or mobility requirement). Therefore, the transport mode is ID-to-ID. Although the transmission process is affected by mobile nodes, our research focuses more on congestion control mechanism rather than mobility support mechanism to ensure the stability for data transmission. The "Next Header" field of the ID header points to our proposed transport protocol. There are two types of packets in ICN, request packets and data packets, which are distinguished by the "Identity" field in the transport layer header. Additionally, the transport layer header includes two important fields, "Offset" and "Length". The "Offset" and "Length" fields of the request packet indicate where the data start to be transmitted and the amount of data that is expected to be received. The receiver calculates the amount of data expected to be received through the congestion control module, and assigns the value to the "Length" field to explicitly inform the sender. Similar to the delayed acknowledgment (ACK) in TCP, the request packets are sent in an aggregated manner, which greatly reduces the overhead caused by the uplink control message. In addition, we design an "Option" field in the transport layer header by TLV (Tag-Length-Value) structure. The information about a segment that is thought to have been lost can be filled into the TLV structure and brought to the sender via the next request packet. In this way, a single request packet can request multiple discrete segments of NDC, which enhances the ability to recover from the loss state. The logic of the sender side is relatively simple. When listening the users' request, the sender only needs to send the corresponding data back according to the ID of NDC, the "Offset" and the "Length" fields in the request packet. Considering that the size of NDC is usually larger than the MTU, the sender needs to split the complete NDC into many small segments and encapsulate a ICN header for them. To ensure the reliable delivery of NDC, we designed an error check mechanism. The fine-grained error check mechanism ensures the reliability of each packet. Specifically, the sender calculates the checksum of the entire packet and fills in the "Checksum" field of the transport layer. The receiver rechecks the field to decide whether to discard it. The concept of an end-to-end connection does not apply in ICN, so our proposed transport protocol is connectionless. There is no handshake throughout the transmission, thus reducing the latency associated with establishing the connection. To ensure the efficient and reliable transmission service for NDC, we also design the retransmission mechanism and the congestion control algorithm.

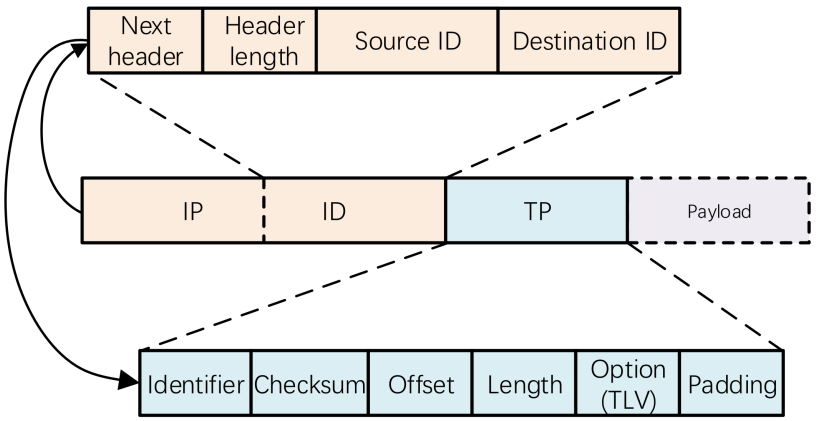

Figure 3. The packet format of ICN header.

3.2.2. Retransmission Mechanism

We design two retransmission mechanisms, including fast retransmission and timeout retransmission. On one hand, fast retransmission enables the transport protocol to recover quickly from the loss state. In TCP, the ACKs are used to acknowledge whether a data packet is received and the fast retransmission is triggered by three duplicate ACKs. In ICN, the fast retransmission mechanism relies on statistics of out-of-order packets because the data packets are used to acknowledge whether the requests are served. In ICTP [15], the fast retransmission is triggered by the arrival of three consecutive out-of-order data packets. In NetInf TP [26], the segment is inferred as lost when the counter of a request entry reaches a predefined value (preferably three, similar to the concept of three duplicate ACKs). Inspired by them, the fast retransmission mechanism we propose is triggered when three out-of-order data packets are received in a row. The receiver immediately generates a new request packet or populates the TLV structure based on the offset and length of the lost segment.

On the other hand, the receiver realizes loss detection by maintaining a timer locally and the timeout retransmission is triggered by the timer expiration. The timeout retransmission depends on the observed RTT and estimation of the RTO. Inspired by TCP, we calculate the RTO in a smooth manner. Readers should note that only if an ordered packet is received, the receiver updates the current RTO. When the timer expires, the receiver immediately re-requests the segment that are thought to have been lost.

3.2.3. Congestion Control Algorithm

During the transmission of NDC, the receiver needs to adjust the request rate according to the feedback signal, including loss signal and marking signal. To be able to coexist with the existing protocol, we designed a window-based congestion control mechanism following the logic of TCP. The receiver maintains a congestion window (*cwnd*) to represent the maximum amount of data allowed to be requested. We refer to some effective congestion algorithms in TCP that have been proven by a large number of theories and experiments, such as reno [45], Cubic [46] and BBR [47], and so on. In this paper, we propose a new window adjustment algorithm, which takes different control operations for loss signal and marking signal. In Section 4.3, we provide more insight into the design of congestion control algorithm.

4. Design of Early Congestion Detection

Considering that it is difficult to detect congestion accurately only relying on the receiver side in ICN [23–25], we propose an early congestion detection mechanism based on an improved AQM algorithm applied in ICN routers. First, we explain the necessity of congestion detection in ICN network. Then, we show the specific steps of early congestion detection mechanism based on an improved AQM algorithm. Finally, we introduce the design of the congestion control algorithm at the receiver side in detail.

4.1. Problem Description

In ICN, the routers not only simply perform forwarding functions, but are also content providers, which makes ICN routers have two types of traffic, external forwarding traffic and internal cache service traffic. With the increase in concurrent services and cache hit rate of ICN routers, the two types of traffic will compete aggressively for bandwidth which is more likely to cause high buffer occupancy and queue fluctuation [13]. Additionally, users' requests can be served by caching routers rather than a fixed remote server, so the traffic in routers is characterized by short bursts. Therefore, it is necessary to propose a congestion detection mechanism applied in ICN routers. Since the classical RED algorithm detects congestion only according to the queue length and its performance is limited by parameters settings [31], we choose a well-studied AQM algorithm, Codel, to judge congestion by monitoring the queuing delay of packets. CoDel simply maintains two parameters, the *target* (default 5 ms) and the *interval* (default 100 ms), and allows the sojourn time of

packets to be higher than the *target* for one *interval*. Specifically, when the sojourn time of a dequeue packet exceeds the *target* but the timeout duration fails to last one *interval*, the router continues to forward the packet without any dropping or marking operation. When the sojourn time of a dequeue packet exceeds the *target* for at least one *interval*, CoDel enters into the dropping state and the router aggressively drops dequeue packet to make the sojourn time converge within the *target*.

Although the Codel algorithm has an excellent performance in the current TCP/IP network [48], we believe that mechanically applying the CoDel algorithm to ICN routers will not yield significant performance gains. The reasons for this include the following two points. First of all, the router still forwards the packet without taking any action in the initial period of congestion, which will cause the receiver to increase the request rate due to the lack of feedback signal, thus aggravating the congestion. Second, the drastic dropping operation in the dropping state may result in the NDC that the lost packet belongs not being able be cached completely, thus reducing the cache success rate. Although the fast retransmission mechanism may solve this problem, the high queuing delay generated by each router is sufficient to release the memory space pre-allocated to the NDC in advance when the network is in a heavy congested state [44].

4.2. Early Congestion Detection Based on Improved AQM Algorithm

The AQM algorithm needs to maintain a low loss rate, high bandwidth utilization and small queuing delay to improve users' QoE (Quality of Experience). As shown in Algorithm 1, we propose an improved AQM algorithm based on CoDel without violating its design principles. According to the queuing delay of packets, we divide the congestion into two phases, early congestion and permanent congestion. The behavior of our improved AQM algorithm depends on the congestion phase. To avoid massive packet loss caused by CoDel frequently entering the dropping state, an active operation should be adopted in the early congestion phase. Considering that the high loss rate will affect the cache success rate and link utilization, we suggest marking instead of dropping at early congestion phase to guide the receiver to adjust the request rate through ECN. Although Codel supports the ECN mechanism, the performance of Codel using the ECN mechanism is insufficient in terms of queue convergence speed [40]. The dequeue packets processing logic of the improved AQM algorithm is shown in Figure 4.

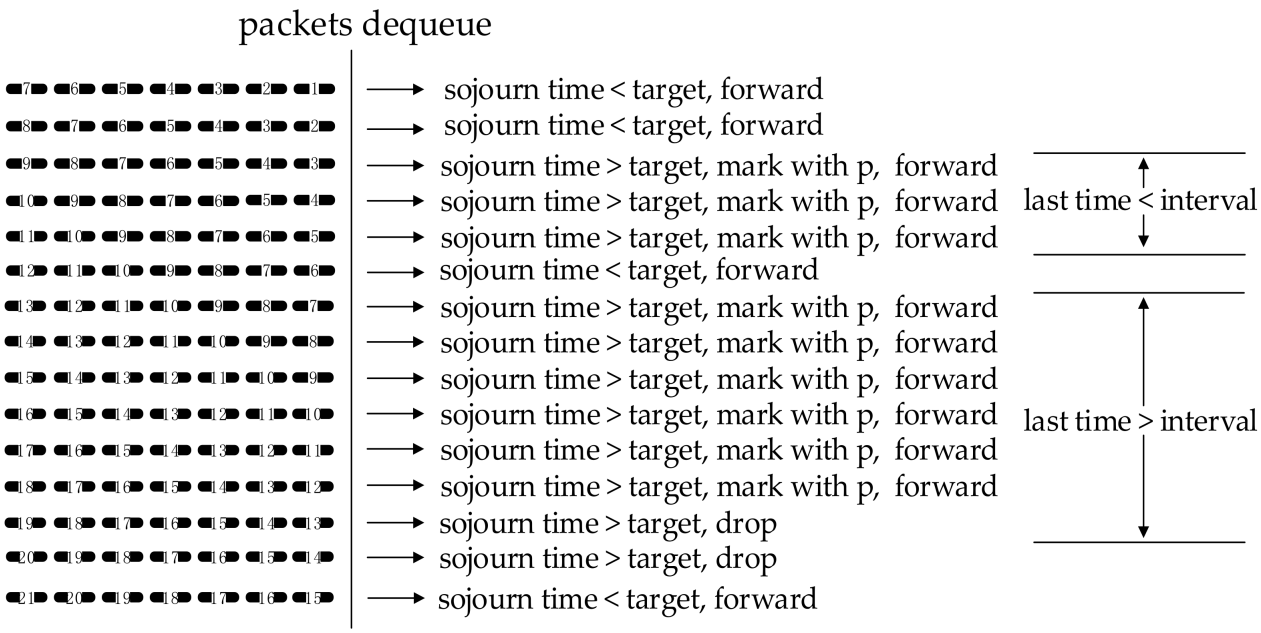

Figure 4. The dequeue packets processing logic of the improved AQM algorithm.

Secondly, when the sojourn time of the dequeue packet exceeds the *target* and the timeout fails to last for an *interval*, it means that the router is in the early congestion state. In addition to the *target* and *interval*, we introduce a probability p to indicate the current congestion degree of the router. Since the sojourn time of the dequeue packet can reflect the congestion degree of the router, the p is linearly positively correlated with the sojourn time. The dequeue packet will be marked with p in advance to avoid frequently entering the dropping state. Additionally, we set a maximum probability P_{max} and a threshold C. When the sojourn time of the dequeue packet exceeds the product of C and *target*, the dequeue packet will be marked with the P_{max}. In [39], the authors who propose CoDel conduct a comprehensive evaluation for CoDel. The results show that the maximum median queuing delay of the packets is about 12.5 ms, which is the product of 2.5 and *target*, under different bandwidths and RTTs. Based on this, we can infer that the congestion is already heavy when the sojourn time reaches 12.5 ms. Therefore, we set the parameter C as 2.5 and P_{max} as 1. When the sojourn time exceeds the product of 2.5 and *target*, all dequeue packets are marked and the receiver reduces the *cwnd* drastically according to the congestion control algorithm so that the router can quickly recover from congestion state. The relationship between congestion degree and queuing delay is shown in Figure 5. The design in which the probability increases linearly with the sojourn time enhances the router's robustness to burst traffic. The calculation of p is shown in Equation (1).

$$p = \begin{cases} 0 & sojourn\_time < target, last\_time < interval \\ \frac{sojourn\_time - target}{(C-1) \times target} P_{max} & C \times target > sojourn\_time > target, last\_time < interval \\ P_{max} & sojourn\_time > C * target \end{cases} \quad (1)$$

where *sojourn_time* is the sojourn time of packet, *last_time* is the duration when the sojourn time of packet exceeds the *target*.

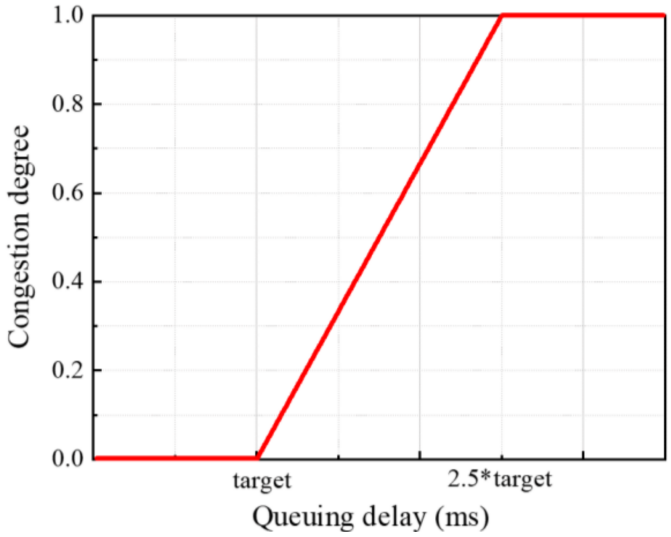

Figure 5. The relationship between congestion degree and queuing delay.

Thirdly, the permanent congestion means that the Codel has entered the dropping state. In this phase, the logic of queue management is taken over by the original Codel algorithm, that is, the dequeue packet will be hard dropped until the sojourn time converges to the *target*. The advantage of this method is that the router can recover from congestion state more quickly.

Because the ICN router has four or even five layers of protocol stacks in our scenario, it can handle packets at the transport layer. To improve the applicability of the improved AQM algorithm, the router marks packets at the transport layer header rather than the network layer header during the early congestion phase. In this way, when the improved AQM algorithm is applied to IP routers, the effect is the same as the original CoDel al-

gorithm. From a theoretical point of view, the usage scenarios of our improved AQM algorithm and the original CoDel algorithm are the same regardless of the wired link or the wireless link, because we have not violated the CoDel's design principle.

Algorithm 1 Improved AQM algorithm

sojourn_time: the sojourn time of the dequeue packet
last_time: the duration of the dequeue packets timeout
p: the marking probability
On departure of every dequeue packet:
if outside the dropping state **then**
  if *last_time* > *interval* **then**
    enter dropping state
    drop the packet
  else
    if *sojourn_time* < *target* **then**
      forward the packet
    else if *target* < *sojourn_time* < *C*target* **then**
      $p \longleftarrow$ Equation (1)
      mark with p and forward the packet
    else
      mark with P_{max} and forward the packet
else if inside the dropping state **then**
  if *sojourn_time* < *target* **then**
    come out dropping state
    forward the packet
  else
    drop the packet

4.3. Congestion Adjustment

The fundamental way to alleviate congestion is to control the number of request packets entering the network. On one hand, when the routers are in the early congestion phase, there is no need to take drastic *cwnd* reduction operations to avoid insufficient link utilization. On the other hand, when the routers are in the permanent congestion phase, a drastic *cwnd* reduction strategy should be adopted to enable the router to quickly recover from the congestion state. As shown in Algorithm 2, we propose a new congestion control algorithm through taking different control operations for loss signal and marking signal, respectively. Specifically, during the initial phase of transmission, the *cwnd* increases according to the slow start algorithm as shown in Equation (2).

$$cwmd = cwnd + 1 \tag{2}$$

As shown in Equation (3), the *cwnd* is adjusted according to the congestion avoidance algorithm when the *cwnd* exceeds the slow start threshold (*ssthresh*).

$$cwnd = cwnd + 1/cwnd \tag{3}$$

When the RTO timer expires, it means that the routers are in the permanent congestion phase. As shown in Equation (4), a drastic *cwnd* reduction method should be adopted at this time.

$$cwnd = cwnd/2 \tag{4}$$

When the marked data packets are received, the receiver should adjust the *cwnd* adaptively instead of mechanically reducing it by half. Inspired by [49], we suggest that the receiver adjusts the *cwnd* according to the congestion degree. The *cwnd* adjustment is shown in Equation (5).

$$cwnd = (1 - \alpha/2) \times cwnd \tag{5}$$

where the parameter α represents the congestion degree. The calculation of α is shown as Equation (6).

$$\alpha = (1 - \omega) \times \alpha + \omega \times F \tag{6}$$

where F is the percentage of marked packets in the previous window received, which is updated once for every window of data (roughly one RTT). ω is the weight given to the new sample relative to the past in the estimation of α, which is an adjustable parameter. According to the suggestion in [49], we set the value of ω as $1/16$.

The proposed receiver-driven congestion control algorithm should be coordinated with the improved AQM algorithm to provide efficient transmission service. Firstly, when the router is not congested, the dequeue packets are forwarded normally and the receiver adjusts the *cwnd* according to slow start algorithm or congestion avoidance algorithm. Secondly, the dequeue packets are marked with the probability p when in the early congestion phase. The receiver detects the percentage of marked packets in the previous window to calculate α. Therefore, the α can reflect the congestion degree of the router to a certain extent and the *cwnd* is completely affected by the congestion degree. When the congestion degree is high, the *cwnd* is almost reduced to half; when the congestion degree is low, the *cwnd* is not affected too much. Thirdly, if CoDel has entered the dropping state, it indicates that the router is in the permanent congestion phase. In this case, the receiver will reduce the *cwnd* to half by detecting packet loss to allow the router to quickly recover from permanent congestion. Readers should note that when the proposed congestion control algorithm is used in conjunction with Codel, RED and other AQM algorithms, its effect is the same as reno algorithm.

Algorithm 2 Congestion Adjustment Algorithm

```
1:  cwnd: congestion control window
2:  ssthresh: slow start threshold
3:  alpha: congestion degree
4:  Initially:
5:    cwnd ⟵ 2
6:    ssthresh ⟵ infinite
7:  New packet received:
8:    if marked packet then
9:      if expire an RTT then
10:       alpha ⟵ Equation (6)
11:       cwnd ⟵ Equation (5)
12:   if non-marked packet then
13:     if cwnd < ssthresh then
14:       cwnd ⟵ Equation (2)
15:     else
16:       cwnd ⟵ Equation (3)
17:  Timeout:
18:    ssthresh ⟵ Equation (6)
19:    cwnd ⟵ssthresh;
```

5. Performance Evaluation

The transmission scheme based on early congestion detection has been clarified. Next, we carry out a series of experiments to evaluate the performance of our proposed transmission scheme in this section. We firstly introduce the experiment parameters. Then, we test the basic performance of our transmission scheme under different conditions, and we conduct an extended experiment in the ICN scenario.

5.1. Experiment Setup

To evaluate the performance of our proposal, we implemented our transmission scheme based on mininet [50]. Mininet is a lightweight network simulator which can work on a real Linux protocol stack by isolating network resources. The receiver-driven transport

protocol and the improved AQM algorithm are integrated into a kernel module and then embedded in the kernel space.

As shown in Figure 6, we performed simulations in a linear topology to evaluate the performance of our proposed transmission scheme. To observe the effect of the transport protocol and the improved AQM algorithm, the linear topology is sufficient for our purposes. The linear topology consists of three ICN routers (R1–R3), one server (S), and 50 clients (C1-C50). ICN client applications are installed on all clients, which are responsible for generating NDC requests and performing congestion control. ICN server applications are installed on the server and ICN routers, which are responsible for sending the corresponding NDC after receiving the requests. Readers should note that different AQM algorithms can be configured on all ICN routers. The default bandwidth and latency between nodes are indicated in Figure 6. The packet size is set to 1500 bytes and the buffer of each ICN router can accommodate 200 packets. The workload is simulated with a catalog of N = 10^3 NDCs which follows a Zipf distribution with a default skewness parameter alpha = 0.8. The NDC size is set to 2 MB by default. The cache capacity of each ICN router is set to 100 NDCs.

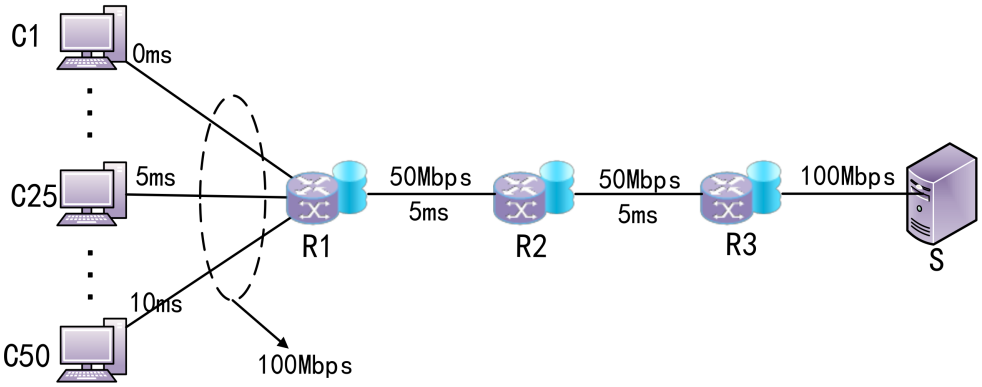

Figure 6. Simulation topology.

To make a comprehensive comparison, The ICN routers are deployed with a passive queue management algorithm, Drop Tail, and several different AQM algorithms, including RED, Codel, and our improved AQM algorithm. These queue management algorithms are used in conjunction with the receiver-driven congestion control to test the performance of our transmission scheme. According to the experience of previous work [30,39], the configuration settings are shown in Table 1.

Table 1. Configuration settings of CoDel and RED.

	Parameter	Value
CoDel	*target*	5 ms
	interval	100 ms
RED	Th_{min}	20%
	Th_{max}	40%
	P_{max}	0.1
	w	0.02

5.2. Basic Performance

We evaluate the basic performance in terms of bandwidth utilization, fairness of multiple flows sharing the same link, buffer occupancy and loss rate. We disable the caching function of the ICN routers and only use the IP-based routing and forwarding capability to verify the basic performance of the transmission scheme. Under this condition, the ICN server can be regarded as content source and the data are transmitted from the ICN server to each ICN client.

5.2.1. Bandwidth Utilization

Bandwidth utilization is an important index to evaluate transmission performance. We test the steady-state throughput (by letting C1 continuously send requests to S) of the transport protocol under different bottleneck bandwidths (by setting the bottleneck bandwidth between routers to 10 Mbps, 20 Mbps, 50 Mbps and 100 Mbps, respectively). Meanwhile, we test TCP steady-state throughput under the same link conditions for a comprehensive comparison. We use the iperf tool to generate TCP traffic on S and transmit it to C1. Readers should note that TCP uses the cubic [46] congestion algorithm. The experimental results are shown in Figure 7. It can be seen that the proposed receiver-driven transport protocol (denoted as TP) can achieve almost the same throughput as TCP under different bottleneck bandwidths. Their throughput difference is no more than 10 KB/s. Moreover, we also test the steady-state throughput when different AQM algorithms are deployed on routers. The throughput remains almost the same when Drop Tail, RED, Codel, and improved AQM are deployed. Their throughputs are 9.54 Mbps, 19.07 Mbps, 47.68 Mbps and 95.37 Mbps, respectively, under different bottleneck bandwidth. This shows that the improved AQM algorithm will not have a negative impact on the bandwidth utilization of the transport protocol.

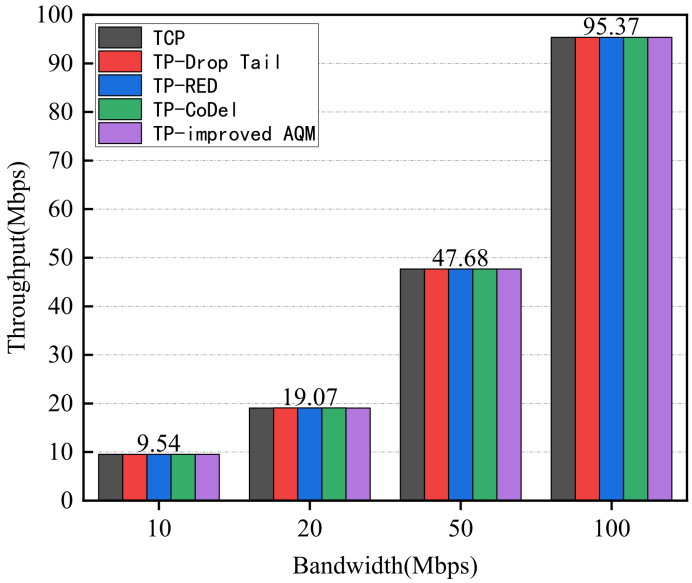

Figure 7. Bandwidth utilization under different bottleneck bandwidth.

5.2.2. Fairness

In ICN, NDC is the basic unit of transmission and storage. A complete content will be split into several NDCs and stored separately in different places, which will cause users to obtain data from multiple places at the same time. Therefore, the transport protocol needs to maintain fairness when there are multiple parallel ICN flows on the bottleneck link. We simulate the situation where multiple ICN flows share a bottleneck link (by letting C1–C5 continuously send requests to S at different moments). Figure 8 shows the throughput of multiple parallel flows sharing a bottleneck link when deploying Drop Tail, RED, Codel, and improved AQM, respectively. From Figure 8a, it can be observed that even if the routers do not deploy any AQM algorithm, all ICN flows can share bandwidth fairly. Moreover, the total bandwidth usage of all flows remains the same when new flows are added, and the remaining flows can quickly take up bandwidth and converge to a fair state when some flows end. As we can see from Figure 8b–d, all flows can maintain better fairness and the throughput of each flow has better stability when deploying RED, Codel, and improved AQM. In conclusion, the experimental results prove that the proposed transport protocol itself has good fairness, even if no AQM is deployed. Meanwhile, the improved

AQM algorithm has almost the same good performance as Codel and RED algorithms in maintaining fairness.

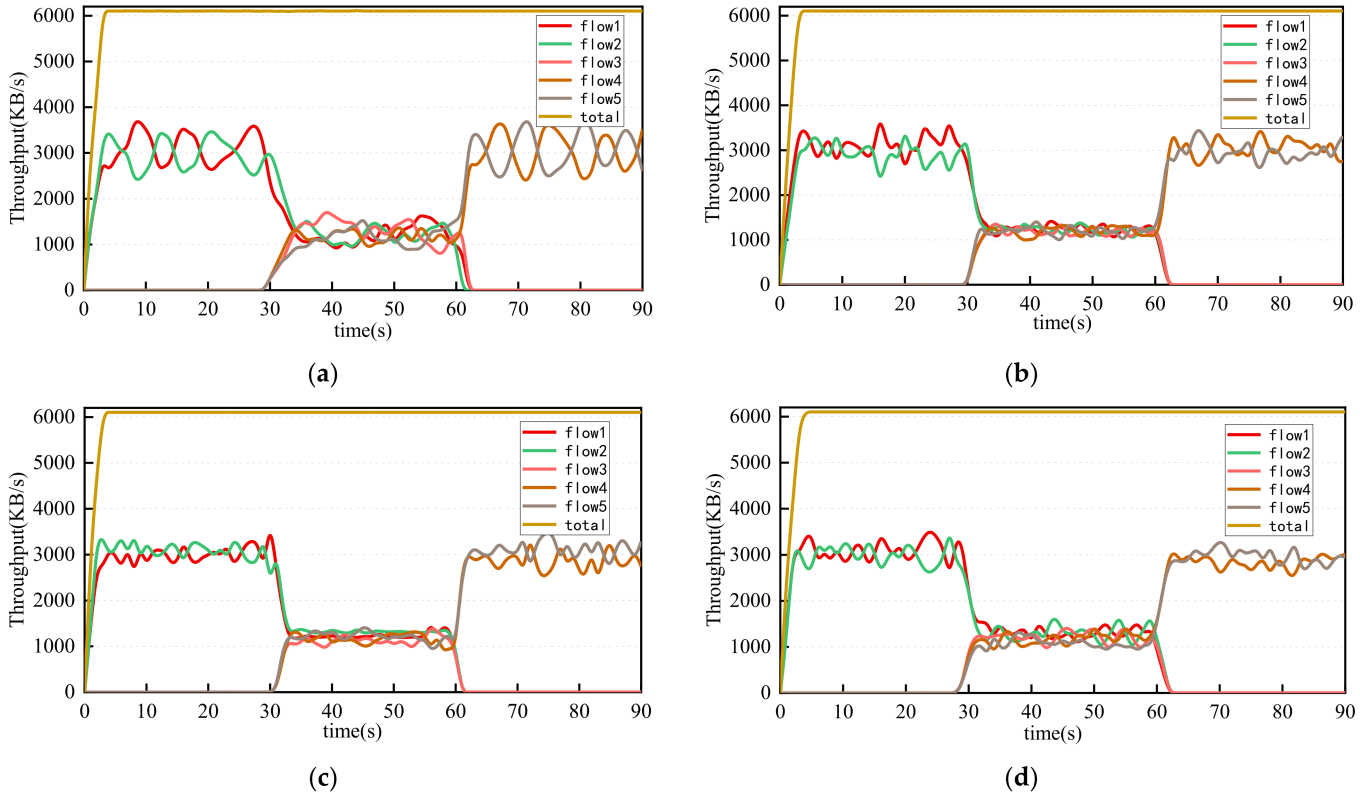

Figure 8. The fairness of the transport protocol in deploying different queue management algorithms: (**a**) Drop Tail; (**b**) RED; (**c**) CoDel; (**d**) Improved AQM.

5.2.3. Buffer Occupancy

The queue formed on routers will cause an extra queuing delay during transmission. In this part, we monitor the instantaneous queue length of routers under a different number of ICN flows to prove the advantages of our transmission scheme. The transmission process is controlled by ICN clients, so each client can maintain a ICN flow by continuously sending request packets to content source. We control the number of ICN flows by varying the number of ICN clients involved in the transmission. Figure 9 shows the queue length over time on R3 when there are 1, 3, 6, 9, 20 and 40 ICN flows in the bottleneck link. Since Drop Tail has no mechanism to actively control the queue, it has the worst effect in controlling queue length. When the number of flows is low, the queue length of the router deploying Drop Tail varies in a zigzag shape, and the queue oscillations are especially obvious. As the number of flows increases, its buffer is gradually filled which will cause the high delay. Moreover, RED and CoDel have a better performance in controlling queue length compared with Drop Tail. As the number of flows increases, RED is slightly better than CoDel in controlling the queue stability. We can also conclude that the improved AQM has the best performance in controlling the queue length and stability when the number of flows is low. As the number of flows increases, the queue stability decreases. This is because the improved AQM algorithm begins to frequently enter the dropping state when a large number of flows exist. Figure 10 shows the variation curve of the average queue length with the number of ICN flows when different queue management algorithms are deployed. The experimental results show that CoDel, RED, and our improved AQM algorithm can control the queue length at a relatively low level compared to the Drop Tail. It is worth noting that the performance of the improved AQM algorithm is significantly better than the other two typical AQM algorithms when the number of flows is low. Additionally,

its performance is also the best even in the presence of a large number of flows. This is because the improved algorithm adopts an early congestion detection mechanism and the receiver adjusts the request rate in time before permanent congestion occurs.

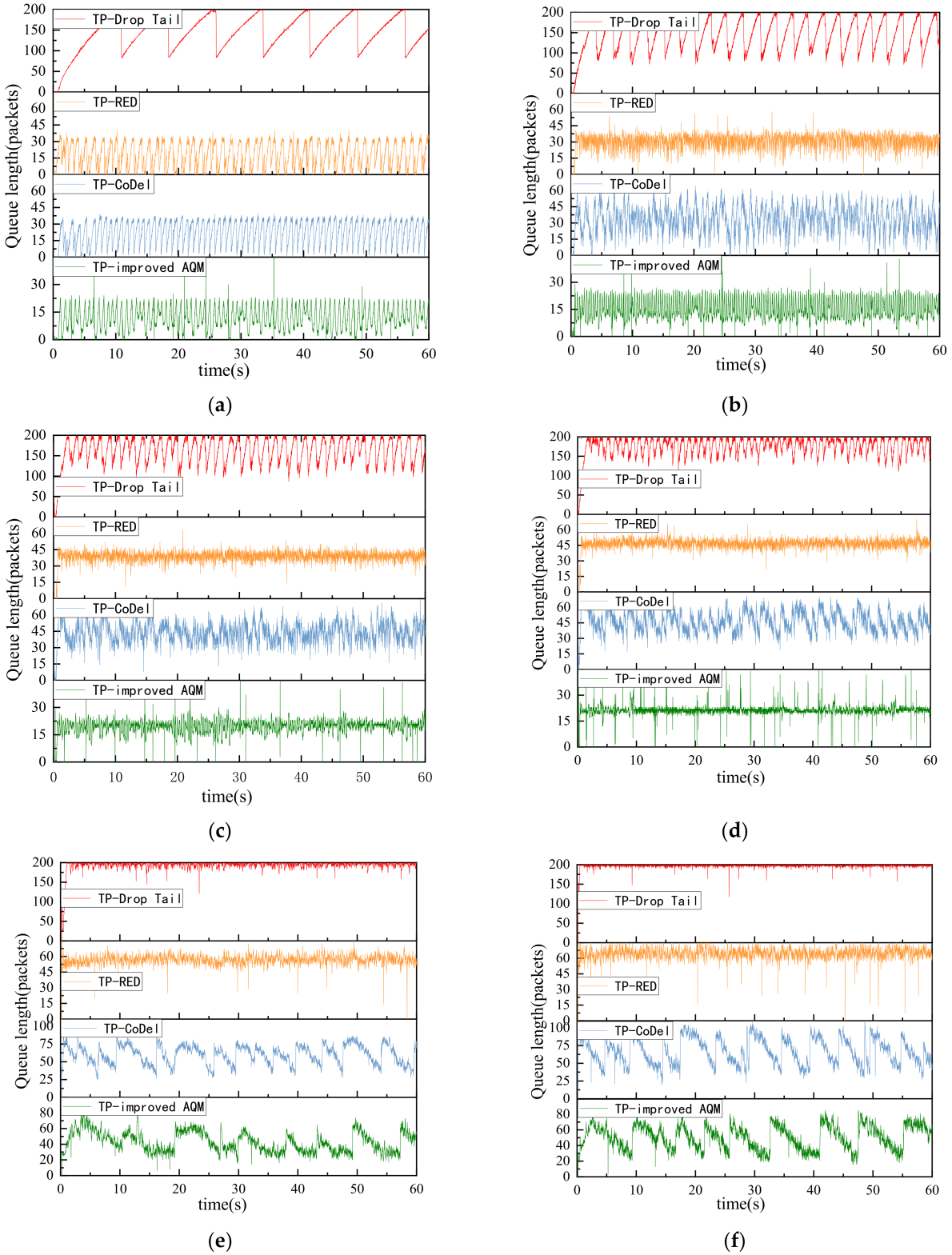

Figure 9. The queue length under different numbers of ICN flows: (**a**) 1 flow; (**b**) 3 flows; (**c**) 6 flows; (**d**) 9 flows; (**e**) 20 flows; (**f**) 40 flows.

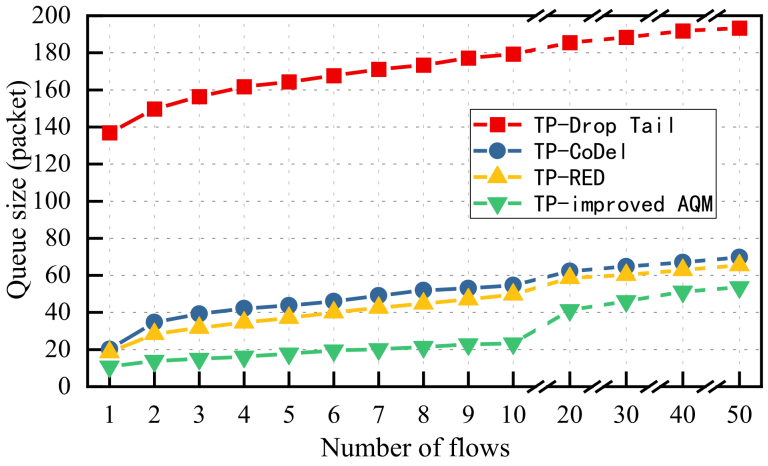

Figure 10. The average queue length under different numbers of ICN flows.

5.2.4. Lost Rate

In this part, we calculate the average packet loss rate of routers under different numbers of ICN flows to illustrate the performance of queue management algorithms in controlling packet loss. Figure 11 shows the packet loss rate of the router using Drop Tail, RED, Codel and improved AQM algorithm, respectively, in the case of different numbers of ICN flows. Although the Drop Tail has a lower loss rate comparing with CoDel and RED algorithms, the long-term high buffer occupancy will cause serious queuing delay. Therefore, Drop Tail is not desirable even if the packet loss rate is relatively low. It can be seen that the CoDel and RED algorithms have a high loss rate. The router deploying the CoDel algorithm continuously drops packets when permanent congestion is detected, which easily leads to obvious queue oscillation and periodic retransmission peak. Similarly, the router deploying the RED algorithm also controls queue length by probabilistically dropping packets so it is difficult to maintain a low packet loss rate. Readers should note that the improved AQM algorithm can significantly reduce the packet loss rate when the number of flows is low, this is because it reduces the frequency of entering the dropping state. As the number of flows increases, it is still better than the other two typical AQM algorithms in controlling packet loss rate.

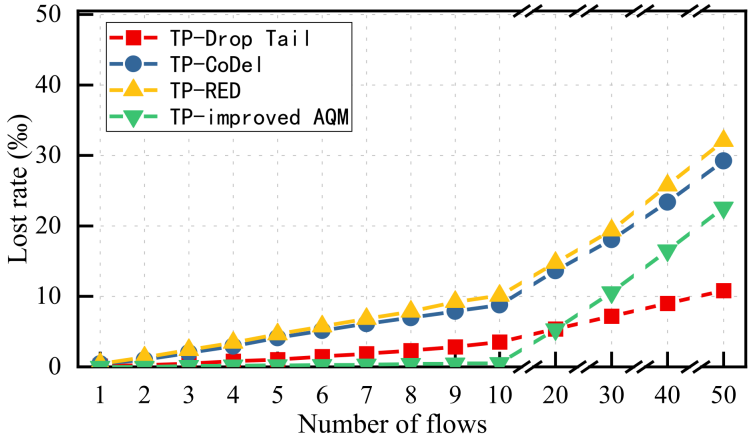

Figure 11. The loss rate under different numbers of ICN flows.

5.3. ICN Scenario Simulation

To simulate the performance of our proposal in ICN scenarios, we enable the caching function of the ICN routers in Figure 6. The ICN server and ICN routers can be regarded as content sources and the data are transmitted from the ICN server or ICN routers to each ICN client under this condition. Similarly, we control the number of ICN flows by

varying the number of ICN clients involved in the transmission. Since all ICN routers can respond to users' requests, the queue can be generated on any ICN router. All ICN routers are deployed with the LCE strategy. At the beginning of the simulation, the cache space of each router is empty. All clients randomly request 500 NDCs. Readers should note that the first 400 NDCs are used to warm up the cache space of routers and the next 100 NDCs are used to obtain statistics for analyzing the proposed transmission scheme. The results are obtained by running the experiments 10 times. We built our simulation scenarios by altering the number of ICN flows (from 1 to 50) and skewness parameter alpha (from 0.6 to 1.3). The default skewness parameter alpha = 0.8 and the default number of ICN flows = 3. We illustrate the effectiveness of our transmission scheme in terms of the average flow completion time (FCT) and average content download delay.

5.3.1. Average FCT

In ICN scenario, a NDC cannot be delivered to the application layer until it is fully received. The FCT represents the transmission efficiency, so we use this indicator to evaluate our proposal. From Figure 12a, it can be concluded that the transport protocol has a longer average FCT when the Codel and RED algorithms are deployed. This is because the fact that the CoDel and RED algorithms frequently drop the packets has a negative impact on goodput. Although the average FCT of the Drop Tail is comparable to that of the improved AQM algorithm in the case of a small number of ICN flows, the additional high queuing delay will make Quality of Service (QoS) of the whole network drop sharply. Compared with the Codel and RED algorithms, the transport protocol has a smaller average FCT when the improved AQM algorithm is deployed, which indicates that the improved AQM algorithm can improve the goodput. Figure 12b shows the average FCT when each client requests 20 NDCs in the case of a large numbers of ICN flows. Similarly, we can conclude that the improved AQM algorithm has a smaller average FCT compared with the Codel and RED algorithms. Figure 13 shows the average FCT for each client requesting 100 NDCs in the presence of three ICN flows. As shown in Figure 13, we can draw a similar conclusion that when transport protocol is used in conjunction with the improved AQM algorithm, the average FCT can be maintained at a minimum level under different skewness.

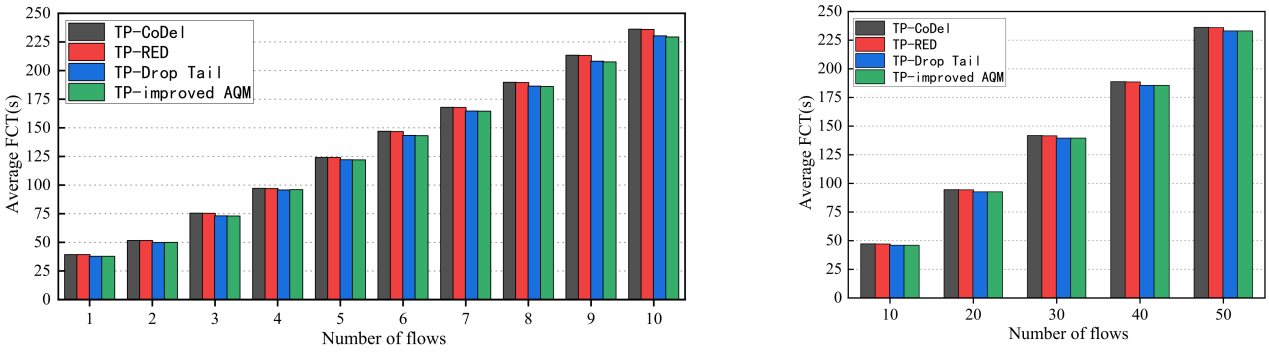

Figure 12. The average FCT under different numbers of ICN flows: (**a**) the case of low number of flows; (**b**) the case of high number of flows.

5.3.2. Content Download Delay

The content download delay is one of the most important indicators of content retrieval performance in ICN scenarios. Although a reasonable caching strategy can also reduce content download delay, our focus is on how to reduce the content download delay through the congestion control mechanism on the determined network path. Figure 14 shows the influence of the number of ICN flows on content download delay. Compared with other methods, we can determine that the transport protocol applying the improved AQM algorithm has the smallest content download delay even if the number of flows grows

high. This is because the improved AQM algorithm can control the queue length to a smaller level. The skewness determines the distribution of the users' requests. The larger the skewness, the higher the cache hit rate of the router and the lower the corresponding content download delay. As shown in Figure 15, with the increase in skewness parameter alpha, the average content download delay is always kept the lowest when transport protocol is used in conjunction with the improved AQM algorithm.

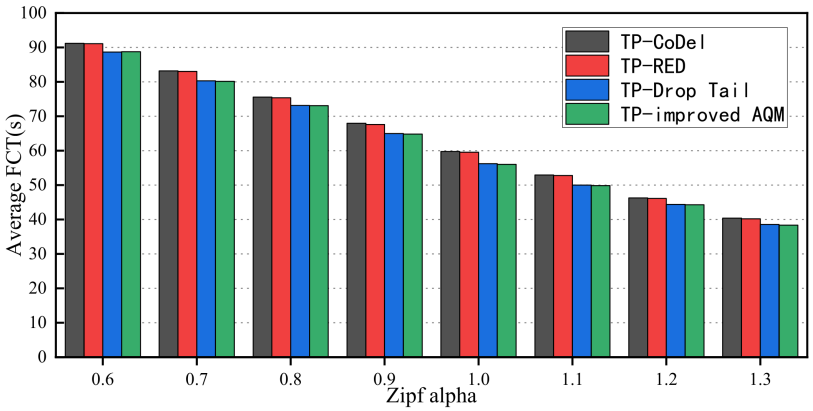

Figure 13. The average FCT under different Zipf alpha.

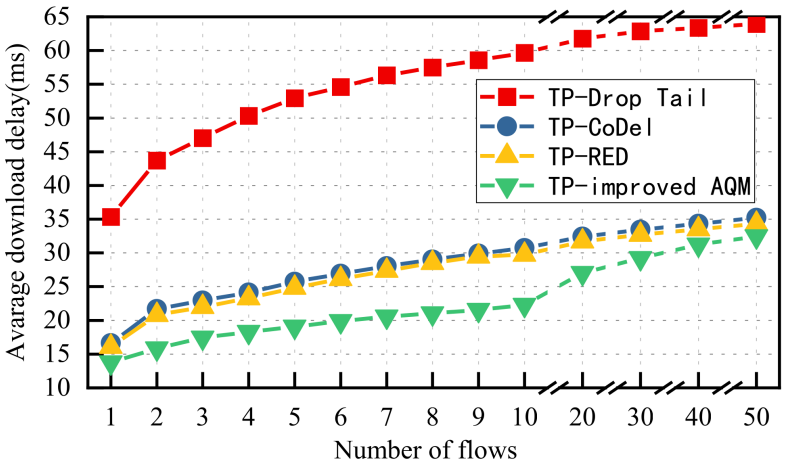

Figure 14. The average content download delay under different numbers of ICN flows.

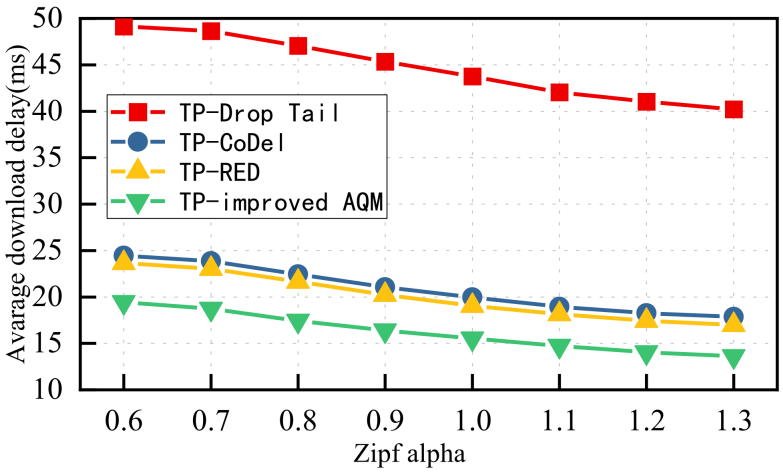

Figure 15. The average content download delay under different skewness.

In summary, our proposed transmission scheme not only has a small average flow completion time but also maintains a low content download delay in ICN scenario. Therefore, our proposed transmission scheme is effective.

6. Conclusions

In this paper, we introduce an effective transmission scheme based on early congestion detection which is used in an IP-compatible ICN architecture. First, we show the receiver-driven transport protocol and describe the key functions in detail, including the transmission process, the protocol stack, the retransmission mechanism, and the congestion control algorithm. Then, we focus on the congestion detection of intermediate routers and propose an improved AQM algorithm according to the queuing delay of packets. Specifically, we define the concepts of early congestion and permanent congestion and the routers take marking and dropping operation, respectively during these two phases. Additionally, a detailed description of the congestion control algorithm, which adopts the gentle and drastic congestion window reduction methods according to the congestion level, is discussed. Finally, the experimental results show that the proposed transmission scheme has an excellent performance in terms of bandwidth utilization, fairness, queue length and loss rate. Meanwhile, when applied in ICN scenarios, the proposed transmission scheme has the shortest average FCT and lowest average download delay.

Future work should focus on two aspects. Firstly, it is necessary to design a reasonable caching strategy to ensure the expected time for downloading content with high probability by combining it with congestion control mechanism. Secondly, due to the extensive content caching supported by routers, data transmission has the characteristics of multipath. We will focus on the design of the multipath transmission mechanism to make full use of the bandwidth of multiple paths and avoid congested paths.

Author Contributions: Conceptualization, Y.X., H.N. and X.Z.; methodology, Y.X., H.N. and X.Z.; software, Y.X.; writing—original draft preparation, Y.X.; writing—review and editing, H.N. and X.Z.; supervision, X.Z.; project administration, X.Z.; funding acquisition, H.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Strategic Leadership Project of Chinese Academy of Sciences: SEANET Technology Standardization Research System Development (Project No. XDC02070100).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We would like to express our gratitude to Jinlin Wang, Yaqin Song, Yifeng Liu for their meaningful support for this work.

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

References

1. Cisco Annual Internet Report (2018–2023) White Paper. Available online: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (accessed on 9 March 2020).
2. Jacobson, V.; Smetters, D.K.; Thornton, J.D.; Plass, M.F.; Briggs, N.H.; Braynard, R.L. Networking named content. In Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, Rome, Italy, 1–4 December 2009; pp. 1–12.
3. Passarella, A. A survey on content-centric technologies for the current Internet: CDN and P2P solutions. *Comput. Commun.* **2012**, *35*, 1–32. [CrossRef]
4. Zhang, L.; Alexander, A.; Jeffrey, B.; Jacobson, V.; Claffy, K.; Crowley, P.J.; Papadopoulos, C.; Wang, L.; Zhang, B. Named data networking. *ACM SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 66–73. [CrossRef]
5. Saxena, D.; Raychoudhury, V.; Suri, N.; Becker, C.; Cao, J. Named data networking: A survey. *Comput. Sci. Rev.* **2016**, *19*, 15–55. [CrossRef]
6. Trossen, D.; Parisis, G. Designing and Realizing an Information-Centric Internet. *IEEE Commun. Mag.* **2012**, *50*, 60–67. [CrossRef]
7. Vahlenkamp, M.; Schneider, F.; Kutscher, D.; Seedorf, J. Enabling ICN in IP networks using SDN. In Proceedings of the 21st IEEE International Conference on Network Protocols (ICNP), Goettingen, Germany, 7–10 October 2013; pp. 1–2.

8. Shailendra, S.; Panigrahi, B.; Rath, H.K.; Simha, A. A novel overlay architecture for information centric networking. In Proceedings of the 2015 Twenty First National Conference on Communications (NCC), Mumbai, India, 27 February–1 March 2015; pp. 1–6.
9. Koponen, T.; Chawla, M.; Chun, B.-G.; Ermolinskiy, A.; Kim, K.H.; Shenker, S.; Stoica, I. A data-oriented (and beyond) network architecture. In Proceedings of the ACM SIGCOMM Computer Communication Review, Kyoto, Japan, 27 August 2007; pp. 181–192.
10. Raychaudhuri, D.; Nagaraja, K.; Venkataramani, A. MobilityFirst: A robust and trustworthy mobility-centric architecture for the future internet. *ACM SIGMOBILE Mob. Comput. Commun. Rev.* **2012**, *16*, 2–13. [CrossRef]
11. Dannewitz, C.; Kutscher, D.; Ohlman, B.; Farrell, S.; Ahlgren, B.; Karl, H. Network of Information (NetInf)—An informationcentric networking architecture. *Comput. Commun.* **2013**, *36*, 721–735. [CrossRef]
12. Wang, J.; Chen, G.; You, J.; Sun, P. SEANet: Architecture and Technologies of an On-site, Elastic, Autonomous Network. *J. Netw. New Media* **2020**, *9*, 1–8.
13. Zeng, L.; Ni, H.; Han, R. The Yellow Active Queue Management Algorithm in ICN Routers Based on the Monitoring of Bandwidth Competition. *Electronics* **2021**, *10*, 806. [CrossRef]
14. Chen, Q.; Xie, R.; Yu, F.; Liu, J.; Huang, T.; Liu, Y. Transport control strategies in named data networking: A survey. *IEEE Commun. Surv. Tut.* **2016**, *18*, 2052–2083. [CrossRef]
15. Detti, A.; Salsano, S.; Cancellieri, M.; Blefari-Melazzi, N.; Pomposini, M. Transport-layer issues in information centric networks. In Proceedings of the 2nd edition of the ICN Workshop on Information-Centric Networking, Helsinki, Finland, 17 August 2012; pp. 19–24.
16. Carofiglio, G.; Gallo, M.; Muscariello, L. ICP: Design and evaluation of an Interest control protocol for content-centric networking. In Proceedings of the 2012 IEEE INFOCOM Workshops, Orlando, FL, USA, 25–30 March 2012; pp. 304–309.
17. Carofiglio, G.; Gallo, M.; Muscariello, L.; Papali, M. Multipath congestion control in content-centric networks. In Proceedings of the 2013 IEEE conference on computer communications workshops (INFOCOM WKSHPS), Turin, Italy, 14–19 April 2013; pp. 363–368.
18. Arianfar, S.; Ott, J.; Eggert, L.; Nikander, P. ConTug: A transport protocol for content-centric networks. In Proceedings of the IEEE International Conference on Network Protocols, Kyoto, Japan, 5–10 October 2010; pp. 1–9.
19. Saino, L.; Cocora, C.; Pavlou, G. CCTCP: A scalable receiver-driven congestion control protocol for content centric networking. In Proceedings of the 2013 IEEE International Conference on Communications (ICC), Budapest, Hungary, 9–13 June 2013; pp. 3775–3780.
20. Rozhnova, N.; Fdida, S. An effective hop-by-hop interest shaping mechanism for ccn communications. In Proceedings of the 2012 IEEE INFOCOM Workshops, Orlando, FL, USA, 25–30 March 2012; pp. 322–327.
21. Yi, C.; Afanasyev, A.; Moiseenko, I.; Wang, L.; Zhang, B.; Zhang, L. A case for stateful forwarding plane. *Comput. Commun.* **2013**, *36*, 779–791. [CrossRef]
22. Yi, C.; Afanasyev, A.; Wang, L.; Zhang, B.; Zhang, L. Adaptive forwarding in named data networking. *ACM SIGCOMM Comput. Commun. Rev.* **2012**, *42*, 62–67. [CrossRef]
23. Ren, Y.; Li, J.; Shi, S.; Li, L.; Wang, G. An explicit congestion control algorithm for named data networking. In Proceedings of the 2016 IEEE conference on computer communications workshops (INFOCOM WKSHPS), San Francisco, CA, USA, 10–14 April 2016; pp. 294–299.
24. Zhang, F.; Zhang, Y.; Reznik, A.; Liu, H.; Qian, C.; Xu, C. A transport protocol for content-centric networking with explicit congestion control. In Proceedings of the 2014 23rd international conference on computer communication and networks (ICCCN), Shanghai, China, 4–7 August 2014; pp. 1–8.
25. Schneider, K.; Yi, C.; Zhang, B.; Zhang, L. A practical congestion control scheme for named data networking. In Proceedings of the 3rd ACM Conference on Information-Centric Networking, Kyoto, Japan, 26–28 September 2016; pp. 21–30.
26. Potys, R.A.; Ali, N.M.; Marsh, I.; Osmani, F. NetInf TP: A receiver-driven protocol for ICN data transport. In Proceedings of the 2015 IEEE 23rd International Symposium on Quality of Service (IWQoS), Portland, OR, USA, 15–16 June 2015; pp. 267–272.
27. Albalawi, A.A.; Garcia-Luna-Aceves, J.J. A Delay-Based Congestion-Control Protocol for Information-Centric Networks. In Proceedings of the 2019 International Conference on Computing, Networking and Communications, Honolulu, HI, USA, 18–21 February 2019; pp. 809–815.
28. Liu, Y.; Zeng, X.; Han, R.; Sun, P. Toward ICN receiver-driven transmission mechanism over WLAN: Implementation and optimization. *Int. J. Innov. Comput. Inf. Control* **2021**, *17*, 853–871.
29. Song, Y.; Ni, H.; Zhu, X. Two-Level Congestion Control Mechanism (2LCCM) for Information-Centric Networking. *Future Internet* **2021**, *13*, 149. [CrossRef]
30. Floyd, S.; Jacobson, V. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.* **1993**, *1*, 397–413. [CrossRef]
31. Floyd, S.; Gummadi, R.; Shenker, S. Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management. Available online: http://www.icir.org/floyd/adaptivered/.html (accessed on 1 August 2001).
32. Floyd, S.; Jacobson, V. Recommendation on Using the Gentle Variant of RED. Available online: www.icir.org/floyd/red/gentle.html (accessed on 10 September 2000).
33. Feng, W.; Shin, K.; Kandlur, D.; Saha, D. The BLUE active queue management algorithms. *IEEE/ACM Trans. Netw.* **2002**, *10*, 513–528. [CrossRef]

34. Misra, V.; Gong, W.-B.; Towsley, D. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. In Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, Stockholm, Sweden, 28 August 2000; pp. 151–160.
35. Hollot, C.V.; Misra, V.; Towsley, D.; Gong, W.-B. On designing improved controllers for AQM routers supporting TCP flows. In Proceedings of the 2001 IEEE INFOCOM Conference on Computer Communications, Anchorage, AK, USA, 22–26 April 2001; pp. 1726–1734.
36. Ding, D.; Zhu, J.; Luo, X. Hopf bifurcation analysis in a fluid flow model of Internet congestion control algorithm. *Nonlinear Anal. Real World Appl.* **2009**, *10*, 824–839. [CrossRef]
37. Zheng, Y.G.; Wang, Z.H. Stability and Hopf bifurcation of a class of TCP/AQM networks. *Nonlinear Anal. Real World Appl.* **2010**, *11*, 1552–1559. [CrossRef]
38. Liu, F.; Guan, Z.; Zhu, G.; Li, T.; Wang, H. Stability analysis and impulsive control of bifurcation and chaos in fluid flow model for TCP/AQM networks. In Proceedings of the 2010 8th World Congress on Intelligent Control and Automation, Jinan, China, 7–9 July 2010; pp. 1257–1261.
39. Nichols, K.; Jacobson, V. Controlling queue delay. *Commun. ACM* **2012**, *55*, 42–50. [CrossRef]
40. Alwahab, D.A.; Laki, S. Ecn-enhanced codel aqm. In Proceedings of the International Conference on Wired/Wireless Internet Communication, Bologna, Italy, 17 June 2019; pp. 157–169.
41. Liao, Y.; Sheng, Y.; Wang, J. A Survey on the Name Resolution Technologies in Information Centric Networking. *J. Netw. New Media* **2020**, *9*, 1–9.
42. Liao, Y.; Sheng, Y.; Wang, J. A deterministic latency name resolution framework using network partitioning for 5G-ICN integration. *Int. J. Innov. Comput. Inf. Control* **2019**, *15*, 1865–1880.
43. Song, Y.; Ni, H.; Zhu, X. Analytical modelling of optimal chunk size for efficient transmission in information-centric network. *Int. J. Innov. Comput. Inf. Control* **2020**, *16*, 1511–1525.
44. Zeng, L.; Ni, H.; Han, R. An Incrementally Deployable IP-Compatible-Information-Centric Networking Hierarchical Cache System. *Appl. Sci.* **2020**, *10*, 6228. [CrossRef]
45. Jacobson, V. Congestion avoidance and control. *ACM SIGCOMM Comput. Commun. Rev.* **1988**, *18*, 314–329. [CrossRef]
46. Ha, S.; Rhee, I.; Xu, L. CUBIC: A new TCP-friendly high-speed TCP variant. *Oper. Syst. Rev.* **2008**, *42*, 64–74. [CrossRef]
47. Cardwell, N.; Cheng, Y.; Gunn, S.C.; Yeganeh, S.H.; Jacobson, V. BBR: Congestion-Based Congestion Control. Commun. *ACM* **2017**, *60*, 58–66. [CrossRef]
48. Raghuvanshi, D.M.; Annappa, B.; Tahiliani, M.P. On the effectiveness of CoDel for active queue management. In Proceedings of the 2013 Third International Conference on Advanced Computing and Communication Technologies (ACCT), Rohtak, India, 6–7 April 2013; pp. 107–114.
49. Alizadeh, M.; Greenberg, A.; Maltz, D.A.; Padhye, J.; Patel, P.; Prabhakar, B.; Sengupta, S.; Sridharan, M. Data center tcp (dctcp). In Proceedings of the ACM SIGCOMM 2010 Conference, New Delhi, India, 30 August–3 September 2010; pp. 63–74.
50. De Oliveira, R.L.S.; Schweitzer, C.M.; Shinoda, A.A.; Prete, L.R. Using mininet for emulation and prototyping software-defined networks. In Proceedings of the 2014 IEEE Colombian Conference on Communications and Computing (COLCOM), Bogota, Colombia, 4–6 June 2014; pp. 1–6.