IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 June 2022, accepted 3 July 2022, date of publication 7 July 2022, date of current version 22 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3189183

RESEARCH ARTICLE

A Novel Active Queue Management Algorithm Based on Average Queue Length Change Rate

CHENGSHENG PAN[1], SONG ZHANG[1], CHEN ZHAO[1], HUAIFENG SHI[1,2], ZHIXIANG KONG[2], AND XIAOSONG CUI[1]

[1]School of Electronics and Information Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China
[2]School of Automation, Nanjing University of Science and Technology, Nanjing 210094, China

Corresponding author: Chengsheng Pan (003150@nuist.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61931004 and Grant 61801073, and in part by the Natural Science Foundation of Jiangsu Province under Grant BK20210641.

ABSTRACT The rapid development of information technology has promoted the transformation of traditional networks into intelligent networks. Huge data traffic is generated by various types of traffic services in the intelligent networks, which can easily lead to network congestion, system instability, and other problems. These problems may incur great requirements and pose challenges for queue management algorithms. Most traditional active queue management (AQM) algorithms judge the congestion level of the network based only on the size of the average queue length while ignoring the network traffic variations. This makes these algorithms difficult to achieve effective improvement of congestion control efficiency. To address this problem, a novel network congestion control algorithm, namely the average queue length and change rate-RED (AC-RED) is proposed in this paper. AC-RED can better relieve network congestion by reconfiguring the packet loss strategy model based on the average queue length change rate. The simulation results of NS2 show that in complex and dynamic network environments, the performance of AC-RED algorithm in the average queue length, packet loss rate, delay, and delay jitter is improved in most load conditions compared with the other five algorithms.

INDEX TERMS Active queue management, network congestion, average queue length change rate, AC-RED, NS2.

I. INTRODUCTION

In recent years, network intelligence has become a prominent feature in the development and transformation of the Internet. Differing from traditional networks, intelligent networks involve a large number of real-time multimedia services, such as web browsing, voice chat, and video conferences. These services generate huge and diverse data traffic [1]–[4]. The buffers of forwarding routers are easily filled up when the network experiences heavy traffic loads, causing further network congestion problems [5]–[7]. End-to-end congestion control provides a direction to think about to alleviate network congestion, and researchers provide many methods to alleviate congestion based on this thought. However, it is difficult for end-to-end congestion control to effectively relieve congestion due to the drawback of lagging feedback congestion information. Therefore, in order to make the effectiveness of the model in ensuring the quality of service (QoS). Congestion control should not only be implemented at the source end but the participation of intermediate nodes of the network should also be considered [8].

Queue management (QM) methods alleviate congestion by managing the queuing of packets in the buffers of intermediate nodes of the network. Generally speaking, according to different packet loss modes, queue management mechanisms can be divided into two categories: passive queue management (PQM) mechanisms and active queue management (AQM) mechanisms. Drop-Tail is a typical PQM algorithm, which was widely used on the Internet because of its simple working principle. But the disadvantages (e.g., ''deadlock'' and ''full queue'') of the Drop-Tail make it likely to lead to increased packet loss, increased delay, and reduced utilization

The associate editor coordinating the review of this manuscript and approving it for publication was Yuan Gao.

of links [9]–[12]. Therefore, new methods need to be proposed to address these issues.

The method is the AQM, proposed by the Internet Engineering Task Force (IETF). The AQM method causes the router buffer to start dropping packets before the average queue length reaches the maximum value and sends congestion feedback information to the source. With this mechanism, sending nodes can reflect before the link cache overflows, effectively reducing or avoiding congestion. Random early detection (RED), as a representative AQM algorithm plays a key role in congestion control. The principle of it is shown in Fig. 1.

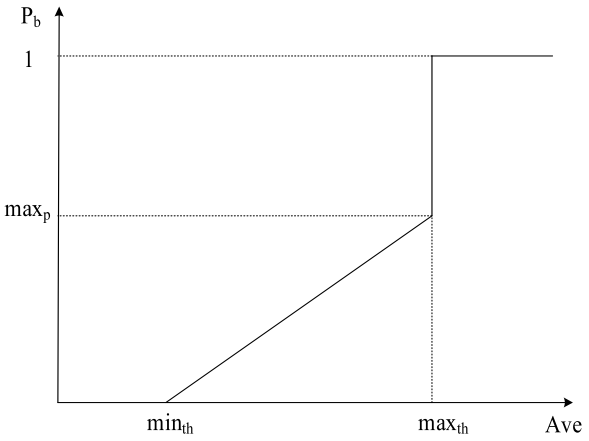


FIGURE 1. Packet loss strategy of RED algorithm.

The router first calculates the average queue length (ave) when it receives newly arrived packets. It determines the packet loss strategy by comparing the current average queue length with the results of the minimum threshold (min_{th}) and the maximum threshold (max_{th}) [13]–[16]. If the average queue length is lower than min_{th}, the drop probability of the queue is 0. This means that all packets arriving at the next moment can be directly transmitted to the routing queue. If the average queue length is between min_{th} and max_{th}, the arriving packets need to be dropped according to the designed packet loss strategy. Finally, if the average queue length is bigger than max_{th}, the drop probability is 1. This means that all packets arriving at the next moment will all be dropped.

Compared with Drop-Tail, RED alleviates congestion of the network. However, there are still several drawbacks to it, such as parameter configuration sensitivity and the phenomenon of global synchronization. These drawbacks are difficult for RED to adapt to dynamically changing network environments. In order to reduce the sensitivity of RED to parameter settings and further improve the congestion management efficiency, researchers have proposed many improved algorithms based on RED, such as ARED [17], URED [18], TRED [19], QARED [20] and SRED [21]. In order to enhance the adaptiveness of the algorithm, ARED dynamically adjusts the maximum drop probability (max_p) according to the change in the average queue length. URED makes better use of the cache space of the router by introducing a new threshold parameter, U_{th}. When the average queue length is in the range $[max_{th}, U_{th}]$, the discard probability is improved from a direct change to 1 in RED to a linear increase to 1. TRED divides $[min_{th}, max_{th}]$ equally into three regions to adapt to different traffic loads; namely $[min_{th}, max_{th}/3 + 2 \times min_{th}/3]$, $[max_{th}/3 + 2 \times min_{th}/3, min_{th}/3 + 2 \times max_{th}/3]$, and $[min_{th}/3 + 2 \times max_{th}/3, max_{th}]$. Each region has a specific drop function, which relieves the network congestion. QARED optimizes the packet loss strategy based on ARED by using a square function in the range $[min_{th}, max_{th}]$, instead of the linear function in ARED. SRED adjusts its drop probability based on traffic load rather than the average queue length. Compared with RED, these improved algorithms improve the performance of the network to some extent. However, the change in traffic is the root cause of the change in links. A serious problem of RED and its improved algorithms is that they ignore the changes in network traffic and judge the congestion of the network only concerning the average queue length. This means that the control efficiency of these algorithms cannot be improved effectively [22].

The AQMRD algorithm has been proposed by Karmeshu *et al.* [23], [24]. Different from the traditional approaches, such as RED and its improved algorithms, AQMRD introduces the average queue length variation rate ($d$ave) as a new parameter, which can reflect the average queue length variation. On this basis, AQMRD further adjusts the packet loss function. These improvements enable AQMRD to stabilize the average queue length and delay at low values, even under frequent network traffic load changes. However, it also has some problems: the packet loss strategy of AQMRD cannot reflect the change of the average queue length well, which leads to the packet loss rate and delay jitter being a bit high. Further, the response speed of the algorithm is also affected.

In order to address the aforementioned problems, we first model the trend of the average queue length variation. Then, a packet loss strategy model is built on this basis, categorically. Finally, a novel active queue management algorithm is proposed based on the system model, namely, the active queue management algorithm AC-RED considering the average queue length change rate. Simulation results using the network simulation software NS2 [25]–[27] show that, in the single-bottleneck link, compared with the other five algorithms, AC-RED algorithm minimizes fluctuations in average queue length under different load conditions; improves packet loss rate and delay performance under most load conditions; and has the best performance in delay jitter under all load conditions. Among the multi-bottleneck link, AC-RED algorithm performs best in packet loss rate, end-to-end average delay, and delay jitter under TCP protocol; AC-RED performs best in packet loss rate, delay jitter, and throughput under TCP and UDP protocols.

The remainder of this paper is structured as follows: Section II discusses the system model of the proposed method. Section III discusses the simulation setup.

In sections IV and V, the simulation results are analyzed. Finally, a conclusion is included in section VI.

II. SYSTEM MODEL

A. QUEUE CHANGE MODEL

The packet loss strategy is the key factor affecting the performance metrics, such as the packet loss rate, delay, delay jitter, and throughput, of AQM algorithms. The packet loss strategy for RED and its variants is based on the assumption that the input traffic characteristics do not vary much over time. However, this does not match the traffic characteristics in the real network environment. Therefore, the average queue length change rate is introduced as a new parameter, in order to reflect the change of network traffic more realistically. The average queue length of AC-RED algorithm and its rate of change are based on the following equations [23].

$$\text{ave(t+1)} = (1 - W_q) \times \text{ave(t)} + W_q \times q\text{(t+1)} \tag{1}$$

$$d\text{ave(t+1)} = (1 - W_q) \times d\text{ave(t)} + W_q \times [q\text{(t+1)} - q\text{(t)}] \tag{2}$$

Eq. (1) is the formula for calculating the average queue length. The average queue length at the current moment is calculated by weighting the average queue length at the previous moment and the instantaneous queue length at the current moment. Eq. (2) is the formula for calculating the average queue length change rate. The average queue length change rate of the current moment is calculated by weighting the average queue length change rate of the previous moment and the difference between the instantaneous queue length of the current moment and the previous moment. The weight parameter W_q symbolizes the sensitivity of the algorithm to the data stream input, the value of it is usually 0.002.

When packets arrive, the time increases by one unit. Then, the average queue length updates and its changing trend will be reflected at this time. Differs from AQMRD in the aspect of defining the rate at which the average queue length reaches the maximum threshold. In AC-RED, we give the principle that when $d\text{ave(t+1)} > 0.005$, the average queue length reaches the maximum threshold quickly. When $0 \leq d\text{ave(t+1)} \leq 0.005$, the average queue length changes smoothly. Furthermore, when $d\text{ave(t+1)} < 0$, the average queue length slows down the rate to reach the maximum threshold. In AQMRD and AC-RED,mid_{th} is updated based on the value of $d\text{ave(t+1)}$. The update about mid_{th} in AQMRD and AC-RED with given in Eq. (3) and Eq. (4), respectively. Compared with AQMRD, the change of mid_{th} in AC-RED is also altered at the same time. In this way, the connection between the packet loss strategy and the average queue length variation rate can be closer and more reasonable. Eq. (5) indicates which packet loss strategy is used in AC-RED by the value of $d\text{ave(t+1)}$ determines.

$$\text{mid}_{\text{th}} = \begin{cases} \text{mid}_{\text{th}} - 1 & d\text{ave(t+1)} > 0 \\ \text{mid}_{\text{th}} & d\text{ave(t+1)} = 0 \\ \text{mid}_{\text{th}} + 1 & d\text{ave(t+1)} < 0 \end{cases} \tag{3}$$

$$\text{mid}_{\text{th}} = \begin{cases} \text{mid}_{\text{th}} - \frac{2}{3} & d\text{ave(t+1)} > 0.005 \\ \text{mid}_{\text{th}}0 \leq & d\text{ave(t+1)} \leq 0.005 \\ \text{mid}_{\text{th}} + \frac{3}{2} & d\text{ave(t+1)} < 0 \end{cases} \tag{4}$$

$$p_b = \begin{cases} p_1 & d\text{ave(t+1)} > 0.005 \\ p_2 & 0 \leq d\text{ave(t+1)} \leq 0.005 \\ p_3 & d\text{ave(t+1)} < 0 \end{cases} \tag{5}$$

B. PACKET LOSS STRATEGY MODEL

AQMRD can reduce the fluctuation range of the average queue length and keep it at a low and stable value in the event of traffic changes, thus leading to lower delay. It can also help source nodes reduce the data sending rate to a certain extent and, thus, improves the network throughput under some load conditions. But in the construction of the packet loss strategy model AQMRD still has several deficiencies, detailed as follows.

When $d\text{ave(t+1)} > 0$, the aggressive packet loss strategy adopted by AQMRD leads to a higher packet loss rate and lower network quality of service. When $d\text{ave(t+1)} \leq 0$, the probability of network congestion is low, but the packet loss probability is imprecise and the router buffers are not fully utilized. Furthermore, the packet loss strategy is not closely linked to the intermediate threshold mid_{th}. And it can be shown that AQMRD links mid_{th} to the packet loss strategy only in the case $d\text{ave(t+1)} > 0$, but does not reflect this link when $d\text{ave(t+1)} \leq 0$.

Therefore, lower delay and more stable average queue length in AQMRD are obtained at the expense of the packet loss rate. So AQMRD still has room for improvement in reducing the packet loss rate, improving the buffer utilization of the router, and reducing the delay jitter. This can be addressed by the following measures, such as optimizing the packet loss strategy when changes in the average queue length are detected, closely linking the packet loss strategy to mid_{th} and congestion level, and making full use of the buffer of the router for the case $d\text{ave(t+1)} = 0$. These measures can effectively improve the efficiency of congestion control.

Through the above analysis, the ideas of URED and TRED are chosen to combine on top of AQMRD. Considering URED, the proposed algorithm adds a larger threshold AC_{th} to max_{th} when $0 \leq d\text{ave(t+1)} \leq 0.005$ (the value of AC_{th} is two-thirds of max_{th}). This serves to make better use of the buffer of routers and improve the efficiency of packet forwarding. Considering TRED, the proposed algorithm processes the packets arriving at the routing node using the segmented packet loss function.

As mentioned above, the average queue length will quickly reach the maximum threshold at $d\text{ave(t+1)} > 0.005$. In this case, the routing nodes are soon in a congested state. The packet loss strategy in AQMRD is shown in Eq. (6). From the equation, it can be seen that the drop probability becomes 1 when $\text{ave} \geq \text{mid}_{\text{th}}$, the packet loss strategy is aggressive at this point.

In order to reduce the packet loss rate, the packet loss strategy needs to be adjusted. When the average queue length is in the range of $[\min_{th}, (\min_{th}+ \mathrm{mid}_{th})/2)$, the packet loss strategy uses the square function. When the average queue length is in the range $[(\min_{th}+ \mathrm{mid}_{th})/2, (\mathrm{mid}_{th}+ \max_{th})/2]$, the packet loss strategy uses power functions. When the average queue length is greater than $(\mathrm{mid}_{th}+ \max_{th})/2$, the packet loss probability becomes 1. This in general ensures that the packet loss probability is greater than RED and less than AQMRD, thus not dropping too many packets. The adjusted drop strategy is shown in Eq. (7).

$$P_1 = \begin{cases} 0 \ \mathrm{ave} < \min_{th} \\ \max_p \times \frac{\mathrm{ave} - \min_{th}}{\mathrm{mid}_{th} - \min_{th}} \ \min_{th} \le \mathrm{ave} < \mathrm{mid}_{th} \\ 1 \ \mathrm{ave} \ge \mathrm{mid}_{th} \end{cases} \tag{6}$$

When $0 \le d\mathrm{ave}(t+1) \le 0.005$, the network traffic trend changes smoothly. The packet loss strategy of the AQMRD algorithm is shown in Eq. (8). In this case, it can be noted that the packet loss strategy used by AQMRD is aligned with RED. The packet loss strategy at this point causes the accurate drop probability to be unavailable when the average queue length is around $\min_{th}$ and $\max_{th}$. The above strategy of using only one drop function around the range $[\min_{th}, \max_{th}]$ cannot reasonably distinguish the degree of congestion. In order to deal with the network congestion more accurately, the piecewise function is used to optimize the discard probability function. The adjusted discard strategy is shown in Eq. (9). The linear function in AQMRD is adjusted using the cubic function and the one third function when the average queue length is in the ranges $[\min_{th}, \mathrm{mid}_{th}]$ and $[\mathrm{mid}_{th}, \max_{th}]$, respectively. In order to improve the utilization of router buffers, we borrow the idea of URED and introduce a new threshold parameter, AC_{th}. When the average queue length is in the range $[\max_{th}, \mathrm{AC}_{th}]$, the packet loss probability does not change to 1 directly, but increases to 1 using a power-of-two function.

It is used because when the average queue length is in the range $[\min_{th}, \mathrm{mid}_{th}]$, the probability of network congestion is small. Therefore, a conservative packet loss strategy is used to reduce the occurrence of packet loss. When the average queue length is in the range $[\mathrm{mid}_{th}, \max_{th}]$, the packet loss probability changes minimally, in order to ensure the stability of the network. When the average queue length is in the range $[\max_{th}, \mathrm{AC}_{th}]$, the network congestion is serious. A relatively aggressive packet loss approach is used, with the packet loss probability range of $[\max_p, 1]$.

$$P_2 = \begin{cases} 0 \ \mathrm{ave} < \min_{th} \\ \max_p \times \frac{\mathrm{ave} - \min_{th}}{\max_{th} - \min_{th}} \ \min_{th} \le \mathrm{ave} < \max_{th} \\ 1 \ \mathrm{ave} \ge \max_{th} \end{cases} \tag{8}$$

$$P_2 = \begin{cases} 0 \ \mathrm{ave} < \min_{th} \\ \frac{\max_p}{2} \times \left(\frac{\mathrm{ave} - \min_{th}}{\mathrm{mid}_{th} - \min_{th}}\right)^3 \ \min_{th} \le \mathrm{ave} < \mathrm{mid}_{th} \\ \frac{\max_p}{2} \times \left[1 + \left(\frac{\mathrm{ave} - \mathrm{mid}_{th}}{\max_{th} - \mathrm{mid}_{th}}\right)^{\frac{1}{3}}\right] \mathrm{mid}_{th} \le \mathrm{ave} < \max_{th} \\ (1 - \max_p) \times \left(\frac{\mathrm{ave} - \mathrm{AC}_{th}}{\mathrm{AC}_{th} - \max_{th}}\right)^{\frac{3}{2}} + 1 \ \max_{th} \le \mathrm{ave} \le \mathrm{AC}_{th} \\ 1 \ \mathrm{ave} > \mathrm{AC}_{th} \end{cases} \tag{9}$$

When the average queue length reaches the maximum threshold at $d\mathrm{ave}(t+1) < 0$, the speed of it slows down, and the probability that the routing node is in the idle state increases. At this time, the packet loss strategy of AQMRD is still shown in Eq. (8). To make better use of the router buffer, a new packet loss strategy is adopted which is more conservative than that for $d\mathrm{ave}(t+1) = 0$ in general. As shown in Eq. (10), in order to stabilize the average queue length within the set range by further reducing packet loss, the average queue length uses a cubic function and a one-half power function in the range $[\min_{th}, \mathrm{mid}_{th}]$ and $[\mathrm{mid}_{th}, \max_{th}]$, respectively.

$$P_3 = \begin{cases} 0 \ \mathrm{ave} < \min_{th} \\ \frac{\max_p}{2} \times \left(\frac{\mathrm{ave} - \min_{th}}{\mathrm{mid}_{th} - \min_{th}}\right)^3 \ \min_{th} \le \mathrm{ave} < \mathrm{mid}_{th} \\ \frac{\max_p}{2} \times \left[1 + \left(\frac{\mathrm{ave} - \mathrm{mid}_{th}}{\max_{th} - \mathrm{mid}_{th}}\right)^{\frac{1}{2}}\right] \mathrm{mid}_{th} \le \mathrm{ave} \le \max_{th} \\ 1 \ \mathrm{ave} > \max_{th} \end{cases} \tag{10}$$

The overall packet loss strategy of AC-RED algorithm is shown in Fig. 2. The advantage of this algorithm is its ability to achieve a more stable average queue length, lower packet loss rate, and lower delay jitter in general.

III. SIMULATION SCENARIO SETUP

This section describes how to set the simulation scenario.

$$P_1 = \begin{cases} 0 \ \mathrm{ave} < \min_{th} \\ \frac{2 \times \max_p}{3} \times \left[\frac{2 \times (\mathrm{ave} - \min_{th})}{\mathrm{mid}_{th} - \min_{th}}\right]^2 \ \min_{th} \le \mathrm{ave} < \frac{\mathrm{mid}_{th} + \min_{th}}{2} \\ \frac{2 \times \max_p}{3} + 2 \times \max_p \times \left[\frac{2 \times \mathrm{ave} - (\mathrm{mid}_{th} + \min_{th})}{\max_{th} - \min_{th}}\right]^{\frac{1}{3}} \ \frac{\mathrm{mid}_{th} + \min_{th}}{2} \le \mathrm{ave} \le \frac{\mathrm{mid}_{th} + \max_{th}}{2} \\ 1 \ \mathrm{ave} > \frac{\mathrm{mid}_{th} + \max_{th}}{2} \end{cases} \tag{7}$$

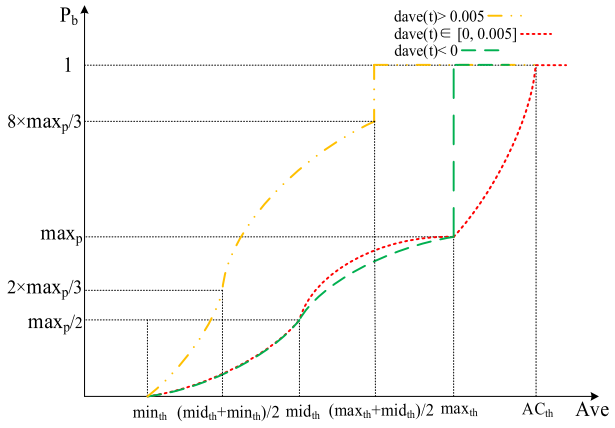


FIGURE 2. Packet loss strategy of AC-RED algorithm.

A. SIMULATION SETUP FOR THE SINGLE-BOTTLENECK LINK

The topology of the single-bottleneck link is shown in Fig. 3. In the figure, routing nodes R_1 and R_2 are chosen to form the bottleneck link. The bandwidth is set to 8 Mbps and the propagation delay is set to 40 ms. S_1 to S_n represents a certain number of FTP transmitting sources, while D_1 to D_n represents a certain number of receiving sources. Transmitting sources are connected to R_1 and receiving sources are connected to R_2. All link bandwidths are set to 100 Mbps and the propagation delay is set to 20 ms. A TCP connection is established between the transmitting source S_n and the corresponding receiving source D_n [28]–[30]. All six algorithms are implemented sequentially on the router R_1 queue. The Drop-Tail algorithm is used for the rest of the queues.

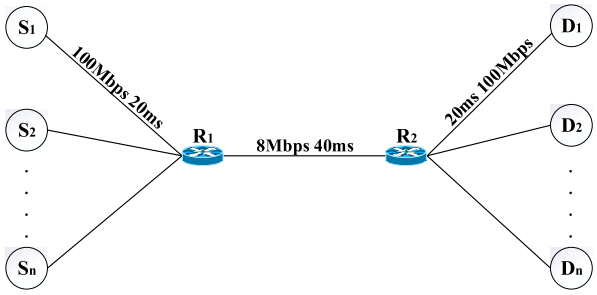


FIGURE 3. Topology of the single-bottleneck link.

Simulation parameters used in the simulation are described below. The value of the minimum threshold min_{th} is 20, which is set to one-third of the maximum threshold max_{th}, in order to obtain good performance. To better utilize the router buffer, the new threshold parameter, AC_{th}, is set to be one and a half times max_{th}. This can increase the tunability of the average queue length and improve the utilization of the router buffer when $0 \leq dave(t+1) \leq 0.005$. Finally, as a rule of thumb, the size of the router buffer (Buffersize) is set to be two times max_{th}.

In the simulation load conditions design session, 50, 75, and 100 FTP sources are simulated. These respectively correspond to conditions where the network is under medium, heavy, and ultra-high traffic loads. In addition, the characteristics of a wide variety of services and bursty traffic in the network also make us add the case of traffic burst in the simulation. In this case, the number of FTP sources is chosen in N = [25, 100]. Specifically, 75 FTP sources are used at the beginning of the simulation; 50 FTP sources are reduced at 25 s and the FTP sources become 25; 25 FTP sources are increased at 50 s and the number becomes 50; and 50 FTP sources are increased again at 75 s, for a total of 100 FTP sources. The simulation time lasts 100 s under all load conditions.

B. SIMULATION SETUP FOR THE MULTI-BOTTLENECK LINK

With the further development and popularization of the Internet, data transmission will involve multiple forwarding devices. And many of the links formed between forwarding devices will often fall into congestion, resulting in the multi-bottleneck link. Therefore, based on the single-bottleneck link, this paper adds a simulation of a multi-bottleneck link to verify the performance of the six algorithms.

Fig. 4 shows a typical multi-bottleneck link consisting of five links connecting six routers and numerous access and output links. The $R_2R_3$and R_4R_5 links are the busiest of the five because they need to forward data not only from the S_n but also from the link with which they are privately connected.

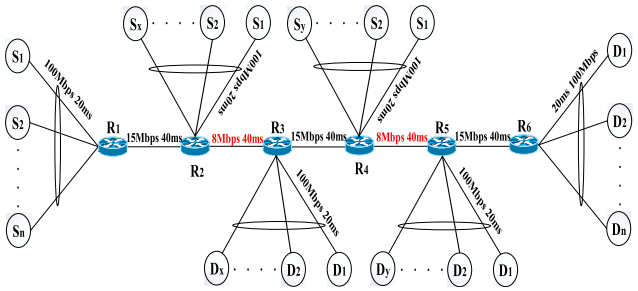


FIGURE 4. Topology of the multi-bottleneck link.

Therefore, R_2R_3 and R_4R_5 are naturally the links for the implementation of AQM methods in this paper. In terms of topology parameter settings, the bandwidth and delay of all access links and output links are the same as those of the single-bottleneck link, that is, 100Mbps and 40ms. Among the five links constituted by routers, the parameters of R_1R_2, R_3R_4, and R_5R_6 links are 15Mbps and 40ms, while those of R_2R_3 and R_4R_5 links are 8Mbps and 40ms, respectively. Simulation parameter settings for the six AQM methods remain the same as those in the single-bottleneck link. Considering that the network is often in congestion. So in the simulation of the multi-bottleneck link, this paper will only simulate the performance of various algorithms when the value of n is 25 and the value of x and y is 50.

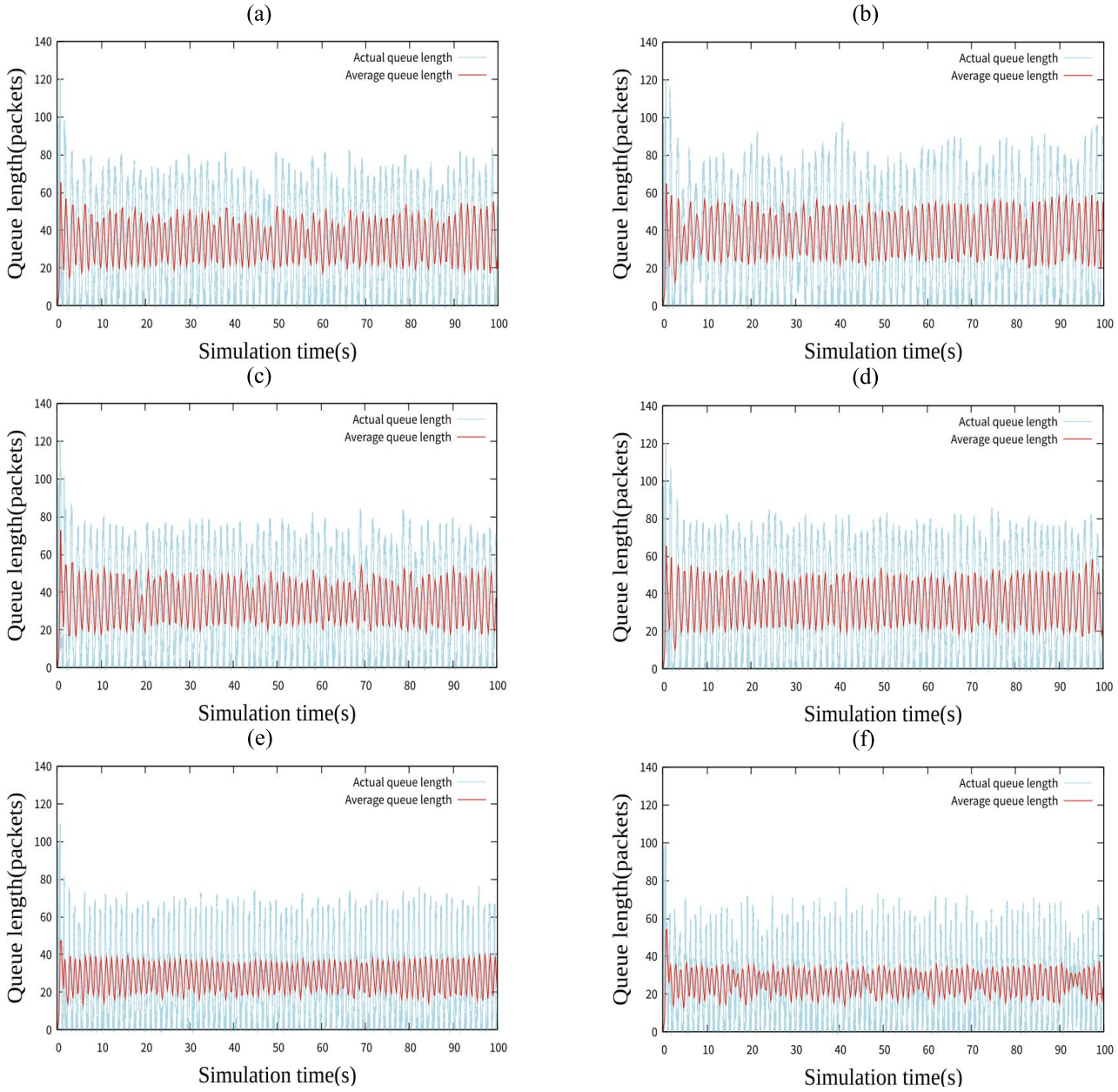


FIGURE 5. Queue length of each algorithm for N = 50: (a) RED, (b) ARED, (c) URED, (d) TRED, (e) AQMRD, (f) AC-RED.

Different services on the network usually have different business requirements, and the transport of traffic inside the link layer sometimes relies on more than just the TCP protocol. Therefore, in the simulation of the multi-bottleneck link, data transmission in two cases will be simulated: the first case is that all senders use TCP protocol to send data; in the second case, five of the sources connect to R_2 and R_4 use UDP.

IV. ANALYSIS OF THE SINGLE-BOTTLENECK LINK SIMULATION

In this section, the performance of the six algorithms will be carefully analyzed.

A. QUEUE LENGTH ANALYSIS

Maintaining a reasonable and stable average queue length can provide a greater capacity to absorb burst packets, thus improving the performance of the network.

Fig. 5 shows the queue length of the six algorithms when the network is under the moderate load condition. At this time, the number of FTP sources is N = 50. In this case, the traditional four algorithms RED, ARED, URED, and TRED basically stabilize the average queue length in the range of $[\min_{th}, \max_{th}]$. In contrast, the average queue length under AC-RED and AQMRD algorithms lie below the minimum threshold for some time. This is mainly because of the relatively low level of network congestion at this time and the

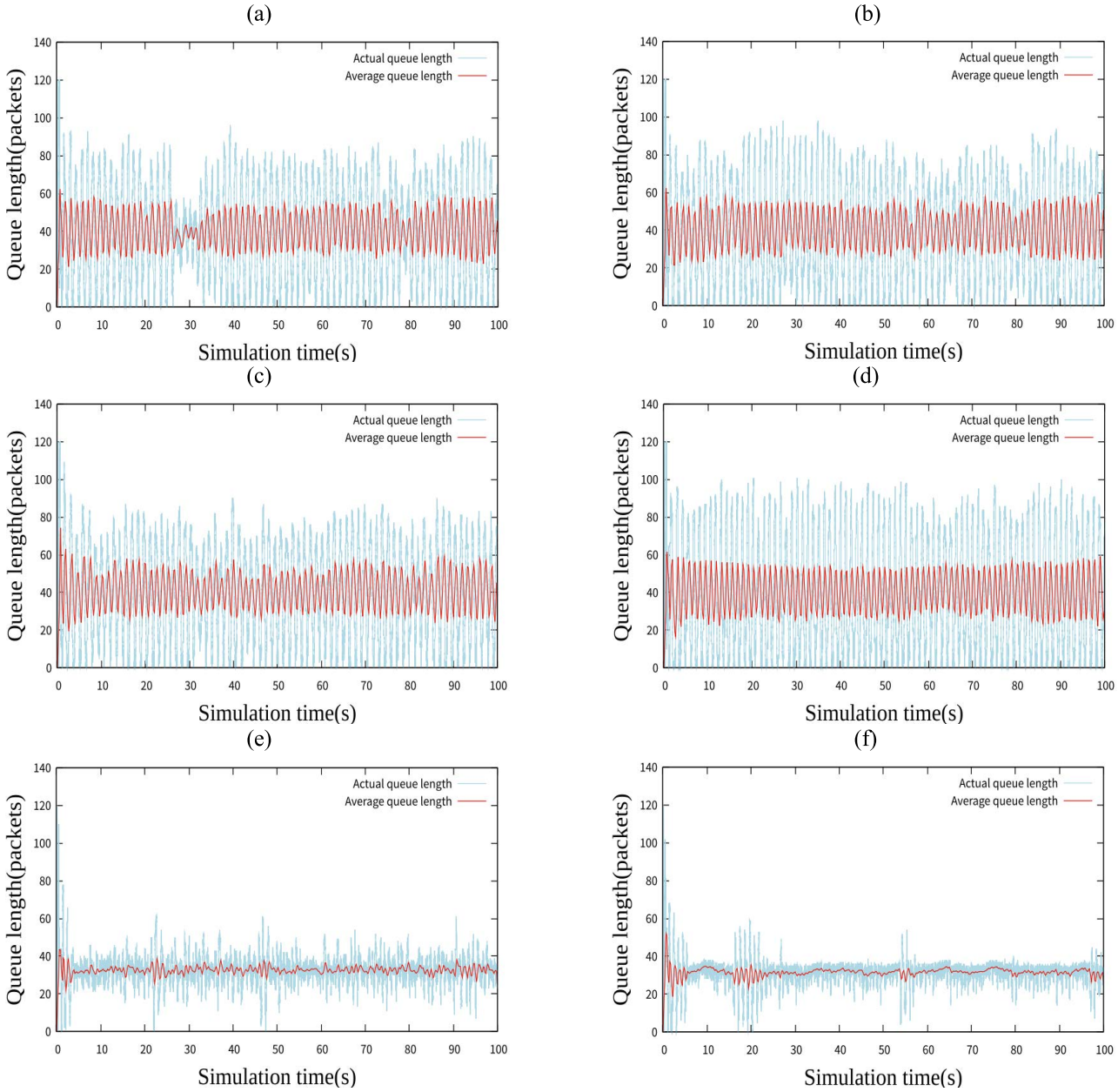


FIGURE 6. Queue length of each algorithm for N = 75: (a) RED, (b) ARED, (c) URED, (d) TRED, (e) AQMRD, (f) AC-RED.

advantage of AC-RED and AQMRD algorithms to detect the trend in the average queue length can enable them to quickly process incoming traffic.

But it can also be seen ifrom the figures that the average queue length fluctuation range of AC-RED algorithm is the smallest, it indicates that the stability of the average queue length under AC-RED algorithm is improved.

Fig. 6 represents the queue length of the six algorithms when the network is under the heavy load condition. At this time, the number of FTP sources is N = 75. In this case, all six algorithms basically stabilize the average queue length in the range of $[\min_{th}, \max_{th}]$. However, the traditional four algorithms have filled up the threshold space and there is no longer enough space to handle the burst traffic that may come. The forwarding efficiency of routing nodes is also reduced. On the contrary, AC-RED and AQMRD algorithms can reduce the occupancy of the threshold space. They stabilize the average queue length around the desired value $0.5 \times (\min_{th} + \max_{th})$, which reduces the fluctuation of the average queue length and improves the forwarding efficiency of routing nodes.

Comparing AC-RED and AQMRD algorithms further, it can be noted from the figures that, the average queue length fluctuation range under AC-RED algorithm is smaller.

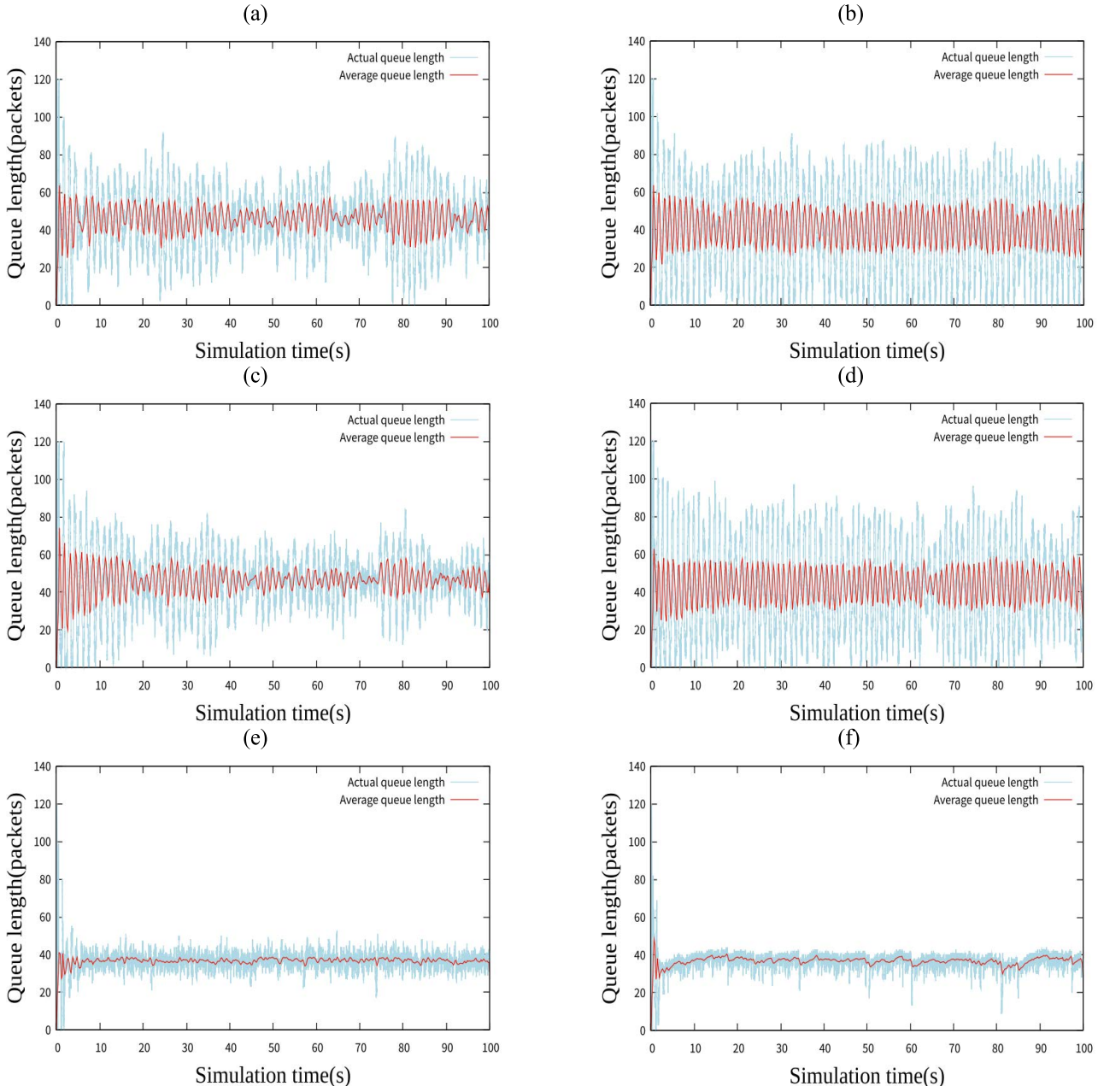


FIGURE 7. Queue length of each algorithm for N = 100: (a) RED, (b) ARED, (c) URED, (d) TRED, (e) AQMRD, (f) AC-RED.

Fig. 7 represents the queue length of the six algorithms when the network is under the ultra-high load condition. At this time, the number of FTP sources is N = 100. In this case, all six algorithms can maintain the average queue length within $[\min_{th}, \max_{th}]$. Compared with the traditional four algorithms, AC-RED and AQMRD can reduce the fluctuation range of the average queue length. As mentioned above, it is because AC-RED and AQMRD can ensure that the router has space to handle data traffic and improve the packet forwarding efficiency.

Comparing AC-RED and AQMRD further, it can be seen from the figures that, the average queue length fluctuation range of AC-RED algorithm is a little smaller. But it must also be noted that the differences between the two algorithms are very small.

Fig. 8 represents the queue length of the six algorithms when the network is under the traffic burst load condition. At this time, the number of FTP sources is N = [25, 100]. Overall, in this case, the average queue length for all six algorithms is below the set minimum threshold when the time is in the range of 25-50 s. The main reason for this situation is the reduction of 50 FTP sources at 25 s during the simulation, the sudden reduction in network traffic allows all algorithms to process the data sent from source

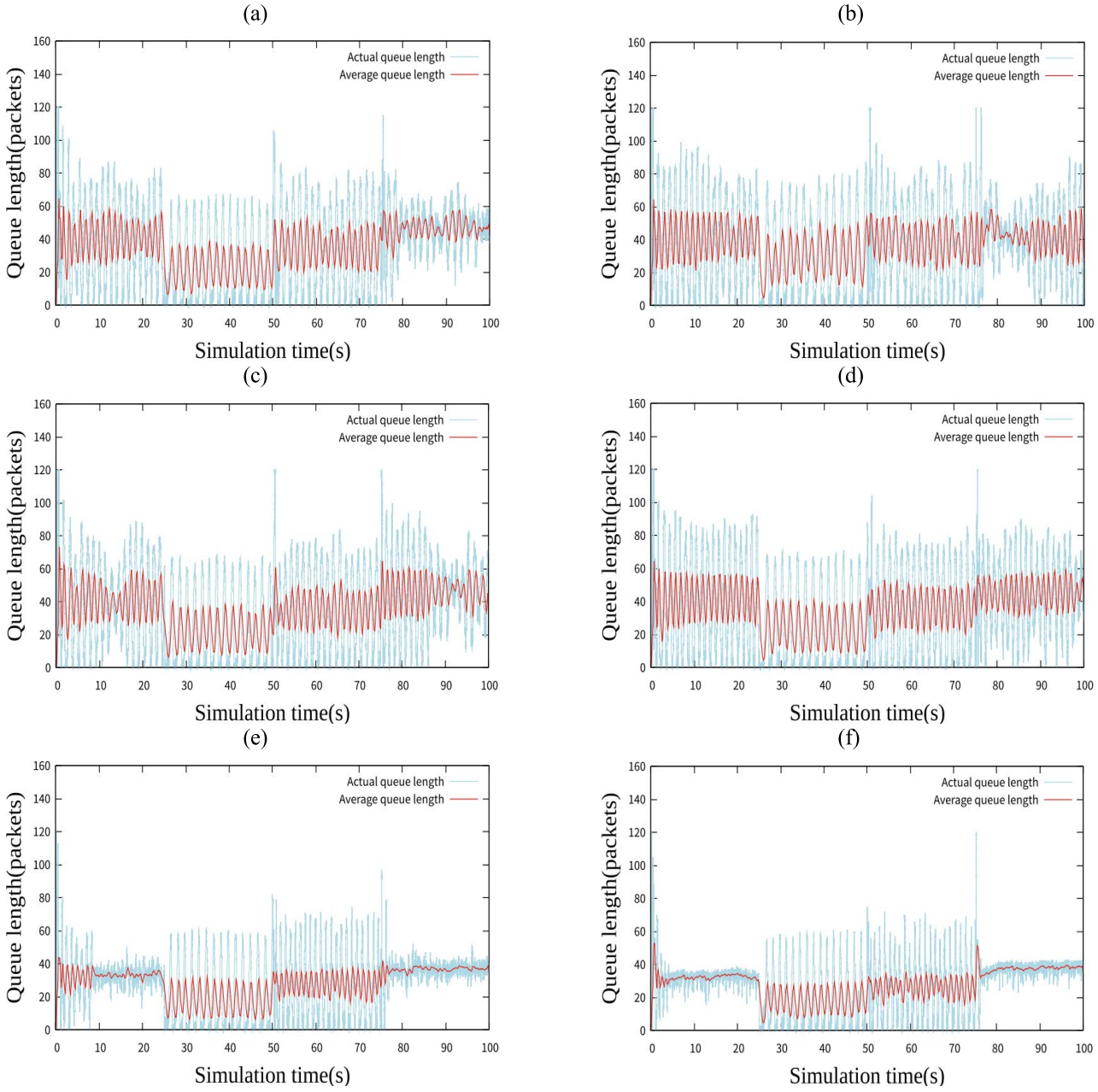


FIGURE 8. Queue length of each algorithm for N = [25,100]: (a) RED, (b) ARED, (c) URED, (d) TRED, (e) AQMRD, (f) AC-RED.

nodes during this time period. After 50 s, the average queue length under all algorithms generally stabilizes in the range [$\min_{th}$, $\max_{th}$] as the source nodes increase. In this case, the fluctuation range of the average queue length is also reduced by AC-RED algorithm, compared with the other five algorithms.

As can be seen from Fig. 5 to Fig. 8, in a single bottleneck link. Compared with the other five algorithms, AC-RED algorithm is able to reduce the fluctuation range of the average queue length whether the network is under medium, heavy, ultra-high, or traffic burst load conditions. This shows that the proposed algorithm has a better performance.

B. PACKET LOSS RATE ANALYSIS

A lower packet loss rate can indicate a reduction in data resource loss and congestion is relieved more effectively.

As can be seen from Fig. 9 and Table 1. The packet loss rate under AC-RED algorithm is significantly lower compared to the other five methods, regardless of whether the network is under heavy, ultra-high, or traffic burst load conditions. Specifically, the packet loss rate under AC-RED algorithm is at least 8.08% lower than the other five algorithms under all three load conditions. This is because AC-RED is able to better adapt to the changing traffic conditions in the network by detecting the trend of the average queue length. Further,

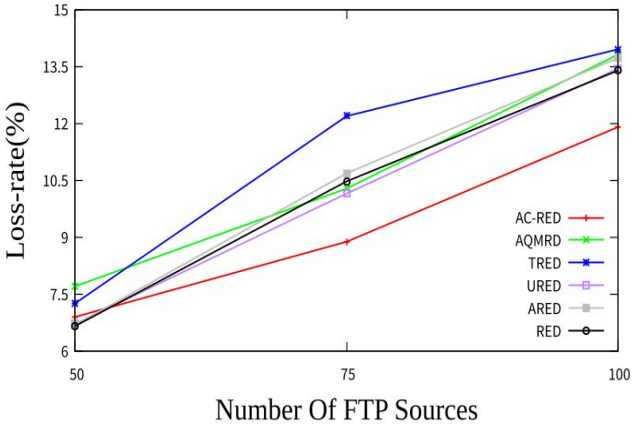


FIGURE 9. Comparison of loss-rate for N = 50, N = 75, N = 100.

TABLE 1. Comparison of loss-rate for N = [25, 100].

Algorithms	Lose/Send	Loss-rate(%)
RED	9145/101306	9.03
ARED	9315/102564	9.08
URED	9172/100806	9.10
TRED	9702/101174	9.59
AQMRD	9481/101369	9.35
AC-RED	8365/100760	8.30

based on the detecting results, AC-RED uses a categorical packet loss strategy to reduce the occurrence of dropped packets arriving at the routing node.

However, it can also be seen that when the network is under the moderate load condition, the packet loss rate under AC-RED algorithm is higher than RED, ARED and URED. This is because in this case, source nodes send relatively less data and the network is not deeply congested. So RED, ARED and URED algorithms can reduce the probability of forced packet loss.

C. END-TO-END AVERAGE DELAY AND DELAY JITTER ANALYSIS

End-to-end average delay and delay jitter are important indicators of network performance. The lower the delay, the better the user experience; the lower the delay jitter, the better the network stability.

The end-to-end average delay and delay jitter of the six algorithms are given in Table 2, where ms and ms^2 are used to denote the units of end-to-end average delay and delay jitter, respectively [31].

In terms of the end-to-end average delay, AC-RED algorithm is the smallest among the six algorithms when the network is under moderate, heavy, and traffic burst load conditions. However, the end-to-end average delay under AC-RED algorithm is higher than that under AQMRD algorithm when the network is under the ultra-high load condition. At this time, the packet loss rate under AC-RED algorithm

TABLE 2. End-to-end average delay and delay jitter.

Algorithms	End-to-end average delay (ms) and delay jitter (ms^2)			
	N=50	N=75	N=100	N=[25, 100]
RED	99.64	101.94	104.75	100.51
	777.59	865.90	765.48	763.35
ARED	101.29	101.84	102.11	101.02
	892.53	894.59	845.73	837.84
URED	99.82	102.08	105.00	100.54
	794.20	875.47	774.36	817.57
TRED	100.35	101.42	103.12	100.66
	853.50	1004.67	893.70	853.99
AQMRD	95.57	97.96	100.10	96.68
	517.10	356.23	417.65	445.40
AC-RED	95.01	97.42	100.13	96.41
	435.24	330.03	417.02	404.22

is very low, so the delay of AC-RED algorithm in this case inevitably increases.

The end-to-end average delay is affected by the size of the average queue length, and AC-RED algorithm is able to keep the average queue length at a low and stable value compared to the other five algorithms. Overall, the performance of the delay under AC-RED algorithm is the best among the six algorithms.

In terms of the delay jitter, regardless of whether the network is under medium, heavy, ultra-high, or traffic burst load conditions, the delay jitter under AC-RED algorithm is the smallest compared to the other five algorithms. This is because AC-RED algorithm is able to detect the trend of the average queue length in real-time and adopt a categorical packet loss strategy for packets arriving at routing nodes after detecting the change.

D. THROUGHPUT ANALYSIS

The throughput is also an important metric for evaluating and measuring the performance of the network. Higher throughput can indicate higher utilization of the bottleneck link.

The throughput of the six algorithms is shown in Fig. 10 and Table 3. The throughput of AC-RED algorithm is the highest of all algorithms when the number of FTP sources is N = 100, the second highest throughput of all algorithms when the number of FTP sources is N = 75 and N = [25, 100], and the fourth highest throughput of all algorithms when the number of FTP sources is N = 50. This indicates that AC-RED algorithm performs poorly in terms of throughput.

The reason for the above situation is that the AC-RED algorithm can more accurately reflect the network congestion to the source node after detecting the traffic trend. And after receiving the congestion information, the source nodes adjust the sending rate of data as well as the total sending volume. Therefore, under some load conditions, the algorithm fails to achieve good performance in throughput.

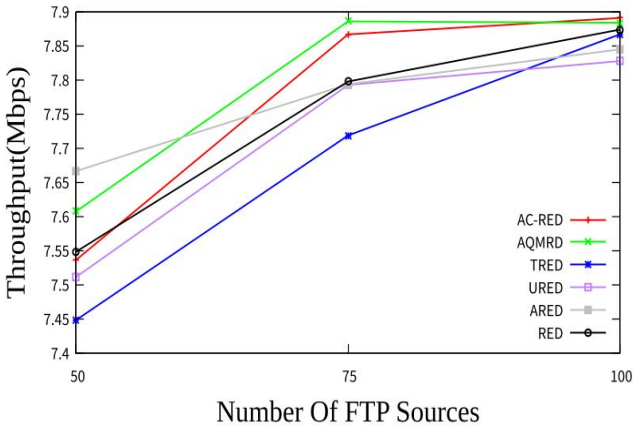


FIGURE 10. Comparison of throughput for N = 25, N = 50, N = 75, N = 100.

TABLE 3. Comparison of throughput for N = [25, 100].

Algorithms	Throughput(Mbps)
RED	7.547
ARED	7.624
URED	7.491
TRED	7.477
AQMRD	7.564
AC-RED	7.607

V. ANALYSIS OF THE MULTI-BOTTLENECK LINK SIMULATION

Since the variations of R_2R_3 and R_4R_5 links are similar, the R_2R_3 link is chosen for the analysis of the simulation results. And because the queue length is qualitatively responsive to the end-to-end average delay and delay jitter, we have done a similar analysis in the single-bottleneck link. Therefore, in the multi-bottleneck link simulation, we will not analyze the queue length.

A. PACKET LOSS RATE ANALYSIS

Table 4 represents the packet loss rates of the six algorithms under TCP protocol, TCP and UDP protocols. From the table, it can be seen that the packet loss rate under AC-RED algorithm is the lowest in either case.

TABLE 4. Comparison of loss-rate.

Algorithms	Loss-rate(%)	
	TCP	TCP and UDP
RED	16.97	15.81
ARED	16.97	16.07
URED	17.01	15.69
TRED	17.61	16.46
AQMRD	12.05	13.74
AC-RED	11.46	12.53

This is because of the categorical packet loss policy used in AC-RED algorithm, and the packet loss policies are all designed based on the variation of the average queue length, reflecting the network congestion. Compared with the traditional algorithms, the probability of being forced to drop the packets due to too many data packets in the queue is greatly reduced.

B. END-TO-END AVERAGE DELAY AND DELAY JITTER ANALYSIS

Table 5 represents the end-to-end average delay as well as delay jitter for the six algorithms under TCP protocol, TCP and UDP protocols. As can be seen from the table, AC-RED algorithm achieves the lowest end-to-end average delay and delay jitter under TCP protocol. The ability of AC-RED algorithm to detect the trend of the average queue length and the use of the classification packet loss policy keeps the data packets queued in the router buffer in a low and stable range, thus reducing the queuing delay and fluctuation of the data packets.

TABLE 5. Comparison of end-to-end average delay and delay jitter.

Algorithms	End-to-end average delay (ms) and delay jitter (ms^2)	
	TCP	TCP and UDP
RED	100.82 1827.39	101.81 1961.50
ARED	101.52 1882.15	102.63 2060.16
URED	100.61 1787.72	101.44 1900.72
TRED	101.42 1900.50	101.96 1966.11
AQMRD	97.20 650.04	104.48 1537.28
AC-RED	96.94 615.29	101.91 1202.54

The average end-to-end delay under AC-RED algorithm is higher than that of URED and RED under TCP and UDP protocols. By analyzing the packets sent under each protocol, we find that fewer packets are sent under AC-RED algorithm using the UDP protocol at this time, and more packets based on the TCP protocol are transmitted, which to a certain extent causes an increase in delay; in terms of delay jitter, the performance of delay jitter under AC-RED algorithm is still the best among all algorithms, which indicates that the data packets under AC-RED algorithm are less volatile in the router buffer when queuing.

C. THROUGHPUT ANALYSIS

Table 6 represents the throughput of the six algorithms under TCP, TCP and UDP protocols. From the table, it can be seen that under TCP protocol, the throughput performance of AC-RED algorithm ranks fifth among the six algorithms, which is almost the worst among the six algorithms. Under TCP and UDP protocols, the throughput performance of AC-RED algorithm is the best among the six algorithms.

One point to note is that high throughput does not mean low delay, and low throughput does not mean high delay. Delay represents the length of time perceived by the application,

while throughput represents the processing efficiency of the entire system. This can be seen from the correspondence between throughput and delay in Table 5 and Table 6.

TABLE 6. Comparison of throughput.

Algorithms	Throughput(Mbps)	
	TCP	TCP and UDP
RED	7.652	7.552
ARED	7.694	7.612
URED	7.627	7.628
TRED	7.548	7.499
AQMRD	7.639	7.612
AC-RED	7.570	7.645

VI. CONCLUSION

In this paper, a novel algorithm, based on the average queue length change rate-called AC-RED has been proposed to improve the efficiency of congestion control. The packet drop strategy of AC-RED is not only dependent on the queue length variation rate but also closely related to the congestion level of the network. Therefore, the proposed method can better stabilize the average queue length and improve the response speed.

The simulation results show that in a single-bottleneck link. The proposed method can achieve the most stable average queue length and the least delay jitter under all load conditions. The lowest delay is obtained under medium, heavy, and traffic burst load conditions. The lowest packet loss rate is obtained under heavy, ultra-high, and traffic burst load conditions. In terms of delay and packet loss, AC-RED algorithm also performs the best of all algorithms, although it does not to obtain the best performance under all load conditions.

In the simulation session about the multi-bottleneck link, this paper simulates the case where some senders use different protocols to send data. In the case of TCP protocol only, the performance of this method is the best among the six algorithms in terms of delay, delay jitter, and packet loss rate; in the case of TCP and UDP protocols, the performance of this method is the best among the six methods in terms of delay jitter, packet loss rate, and throughput. This demonstrates the effectiveness of the improved method.

Of course, the proposed algorithm also has disadvantages. The performance under the multi-bottleneck link needs to be improved; the performance of the throughput also needs to be improved, because the poor throughput performance is not only found in the multi-bottleneck link, but also in the single-bottleneck link. These are where we need to focus on improving in the future.

REFERENCES

[1] I. Javed, X. Tang, K. Shaukat, M. U. Sarwar, T. M. Alam, I. A. Hameed, and M. A. Saleem, "V2X-based mobile localization in 3D wireless sensor network," *Secur. Commun. Netw.*, vol. 2021, pp. 1–13, Feb. 2021.

[2] H. Abdel-Jaber, "An exponential active queue management method based on random early detection," *J. Comput. Netw. Commun.*, vol. 2020, pp. 1–11, May 2020.

[3] C. Pan, Y. Wang, H. Shi, J. Shi, and R. Cai, "Network traffic prediction incorporating prior knowledge for an intelligent network," *Sensors*, vol. 22, no. 7, p. 2674, Mar. 2022.

[4] M. A. Saleem, Z. Shijie, and A. Sharif, "Data transmission using IoT in vehicular ad-hoc networks in smart city congestion," *Mobile Netw. Appl.*, vol. 24, no. 1, pp. 248–258, Jan. 2019.

[5] G. F. Liu and W. Zhang, "Path congestion control algorithm for large delay networks under active queue management," *Comput Simulat.*, vol. 38, no. 3, pp. 268–271, Mar. 2021.

[6] S. Duan, V. Shah-Mansouri, Z. Wang, and V. W. S. Wong, "D-ACB: Adaptive congestion control algorithm for bursty M2M traffic in LTE networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9847–9861, Dec. 2016.

[7] J. Szyguła, A. Domański, J. Domańska, D. Marek, K. Filus, and S. Mendla, "Supervised learning of neural networks for active queue management in the internet," *Sensors*, vol. 21, no. 15, p. 4979, Jul. 2021.

[8] S. Patel, A. Gupta, and M. Singh, "A new active queue management algorithm: Altdrop," in *Proc. ICACCCN, Greater Noida*, Uttar pradesh, India, 2018, pp. 124–127.

[9] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, and L. Peterson, "Recommendation on queue management and congestion avoidance in the internet," IETF, Tech. Rep. RFC 2309, Apr. 1998.

[10] M. A. Saleem, Z. Shijie, M. U. Sarwar, T. Ahmad, A. Maqbool, C. S. Shivachi, and M. Tariq, "Deep learning-based dynamic stable cluster head selection in VANET," *J. Adv. Transp.*, vol. 2021, pp. 1–21, Jul. 2021.

[11] M. M. Abualhaj, M. M. Al-Tahrawi, A. H. Hussein, and S. N. Al-Khatib, "Fuzzy-logic based active queue management using performance metrics mapping into multi-congestion indicators," *Cybern. Inf. Technol.*, vol. 21, no. 2, pp. 29–44, Jul. 2021.

[12] M. A. Saleem, S. Zhou, A. Sharif, T. Saba, M. A. Zia, A. Javed, S. Roy, and M. Mittal, "Expansion of cluster head stability using fuzzy in cognitive radio CR-VANET," *IEEE Access*, vol. 7, pp. 173185–173195, 2019.

[13] J. X. Ren, M. Q. Jiang, and C. H. Wen, "RED algorithm based on S-type function dynamic adaptive improvement," *High Technol. Commun.*, vol. 29, no. 5, pp. 449–454, May 2021.

[14] H. Hu and L. Yao, "Improvement for congestion control algorithms under DDoS attacks," in *Proc. Int. Conf. Comput. Intell. Softw. Eng.*, Hubei, China, Dec. 2009, pp. 1–4.

[15] O. Elloumi and H. Afifi, "RED algorithm in ATM networks," in *Proc. IEEE ATM Workshop*, Lisboa, Portugal, May 1997, pp. 312–319.

[16] Z. Liu, J. Sun, S. Hu, and X. Hu, "An adaptive AQM algorithm based on a novel information compression model," *IEEE Access*, vol. 6, pp. 31180–31190, 2018.

[17] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: An algorithm for increasing the robustness of RED's active queue management," Tech. Rep., Aug. 2001.

[18] C. M. Patel, "URED: Upper threshold RED an efficient congestion control algorithm," in *Proc. 4th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Tiruchengode, India, Jul. 2013, pp. 1–5.

[19] C.-W. Feng, L.-F. Huang, C. Xu, and Y.-C. Chang, "Congestion control scheme performance analysis based on nonlinear RED," *IEEE Syst. J.*, vol. 11, no. 4, pp. 2247–2254, Dec. 2017.

[20] L. Xue, "Implementation of an improved ARED congestion control algorithm," *Comput. Technol. Develop.*, vol. 38, no. 3, pp. 117–121, Mar. 2020.

[21] A. K. Paul, H. Kawakami, A. Tachibana, and T. Hasegawa, "An AQM based congestion control for eNB RLC in 4G/LTE network," in *Proc. IEEE Can. Conf. Electr. Comput. Eng. (CCECE)*, Tamilnadu, India, May 2016, pp. 1–5.

[22] A. Adamu, V. Shorgin, S. Melnikov, and Y. Gaidamaka, "Flexible random early detection algorithm for queue management in routers," in *Proc. ICCCN*, Honolulu, HI, USA, 2020, pp. 196–208.

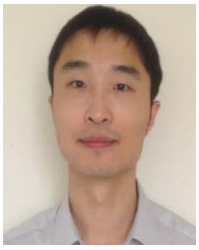
[23] A. Karmeshu, S. Patel, and S. Bhatnagar, "Adaptive mean queue size and its rate of change: Queue management with random dropping," *Telecommun. Syst.*, vol. 65, no. 2, pp. 1–15, Jun. 2017.

[24] S. Bhatnagar, S. Patel, and Karmeshu, "A stochastic approximation approach to active queue management," *Telecommun. Syst.*, vol. 68, no. 1, pp. 89–104, May 2018.

[25] T. Issariyakul and E. Hossain, "Introduction to network simulator NS2," in *Introduction to Network Simulator NS2*. Boston, MA, USA: Springer, 2012, pp. 21–40. [Online]. Available: https://link.springer.com
[26] Z. H. Ke, R. X. Cheng, and D. X. Deng, "NS2 simulation experiment: Multimedia and wireless network communication," in *NS2 Simulation Experiment: Multimedia and Wireless Network Communication*. Beijing, China: Electronics Industry, 2009, pp. 141–164. [Online]. Available: https://xueshu.-baidu.com
[27] Y. H. Wu and G. Z. Liu, "Function extension based on NS-2 network simulation protocol," *Comput. Technol. Develop.*, vol. 19, no. 12, pp. 63–66, Dec. 2009.
[28] C. Su, G. Jin, X. Jiang, and J. Niu, "Research on fair and low delay active queue management algorithm," *J. Commun.*, vol. 38, no. 5, pp. 199–206, May 2017.
[29] H. J. Lei, P. Xu, and Y. J. Xu, "Research on TCP teaching based on NS2 simulation," *Sci. Educ. Literature Rev.*, vol. 12, no. 6, pp. 69–71, Jun. 2017.
[30] I. Javed, X. Tang, M. A. Saleem, M. U. Sarwar, M. Tariq, and C. S. Shivachi, "3D localization for mobile node in wireless sensor network," *Wireless Commun. Mobile Comput.*, vol. 2022, pp. 1–12, Mar. 2022.
[31] L. R. Tang and Y. Tan, "Adaptive queue management based on the change trend of queue size," *KSII Trans. Int. Inf. Syst.*, vol. 13, no. 3, pp. 1345–1362, Mar. 2019.

CHENGSHENG PAN was born in 1962. He received the B.S. and M.S. degrees from the Nanjing University of Science and Technology, in 1984 and 1987, respectively, and the Ph.D. degree from Northeastern University, in 2001. From 1988 to 1989, he was a Co-Researcher at the Institute of Mathematics, Chinese Academy of Sciences. Since 1989, he has been an Assistant Professor with Shenyang Ligong University. He is currently a Professor with the Nanjing University of Information Science and Technology and a part-time Ph.D. Tutor with the Nanjing University of Science and Technology. He is the author of three books and more than 150 articles. His research interests include intelligent network traffic theory and key technologies.

SONG ZHANG was born in 1996. He received the B.S. degree from the Changzhou Institute of Technology, Changzhou, China, in 2020. He is currently pursuing the M.S. degree in electronic and communication engineering with the Nanjing University of Information Science and Technology. His research interest includes flow control.



CHEN ZHAO received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2011, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2016. He was a Research Scientist at the Temasek Laboratory, National University of Singapore, from August 2015 to December 2017. He is currently with the Nanjing University of Information Science and Technology. His research interests include antenna design and wireless communications.

HUAIFENG SHI was born in 1989. He received the B.S. and M.S. degrees from the Nanjing University of Science and Technology, Nanjing, Jiangsu, China, in 2007 and 2011, respectively, where he is currently pursuing the Ph.D. degree. He was a Lecturer at Dalian University, in 2019. He is currently working as a Lecturer with the Nanjing University of Information Science and Technology. His main research interests include space-ground integrated network technology and heterogeneous link aggregation methods in wireless networks.

ZHIXIANG KONG received the M.S. degree from the Information College, Dalian University, Liaoning, China, in 2018. He is currently pursuing Ph.D. degree with the Nanjing University of Technology, Nanjing. His research interests include intelligent network traffic theory and its key technologies.

XIAOSONG CUI received the M.S. degree in electronic and communication engineering from the Jiangsu University of Science and Technology, Zhenjiang, China, in 2020. He is a Research Assistant with the Nanjing University of Information Science and Technology, Nanjing. His research interests include intelligent network traffic theory and its key technologies.