Simulation and Implementation of Adaptive Fuzzy PID

Xiaoping Yang
State Key Laboratory of Automotive Simulation and Control, Jilin University, Changchun 130025, China
College of Communication Engineering, Jilin University, Changchun 130012, China
Email: yxp@jlu.edu.cn.

Chunfeng Song, Jiwei Sun, and Xiaojing Wang
College of Communication Engineering, Jilin University, Changchun 130012, China
Email: {songcf12, sunjw13, wangxj13}@mails.jlu.edu.cn.

Abstract—With the exponential growth of the Internet traffic, congestion problems become severe. To solve these severe problems, the Internet Engineering Task Force (IETF) recommends implementing the Active Queue Management (AQM) mechanism on the Internet routers. This paper introduces a robust AQM algorithm, called Adaptive Fuzzy Proportional Integral Derivative (AF-PID), which adaptively tunes the parameters to overcome the sensitivity of parameters settings for dynamic Internet environments. It’s more important that we have operated comparison experiments between Droptail, RED, PID and AF-PID algorithm under seven different scenarios by NS2 simulation software. The experimental results definitely proved the performance of the AF-PID algorithm is of great superiority. Furthermore, we have built up a network test bed with IPCOP as the router successfully, and embedded with the AF-PID algorithm, which once again proved that AF-PID algorithm is not only easy to implement on real network, but also consistent with the simulation results under the same scenario.

Index Terms—AQM; Fuzzy Control; Adaptive Fuzzy PID; Congestion Control

I. INTRODUCTION

Congestion control is very important, especially for today’s Internet environment, consumers are more focusing on the quality of experience for Internet. Therefore the congestion control is of high importance, so that the Internet Engineering Task Force (IETF) recommends strongly that the Active Queue Management [1] (AQM) schemes to be implemented on each Internet router [2]. The aim of AQM schemes is to stabilize the queue lengths, reduce the end-to-end delay, minimize the packet drop rate, and also to avoid the lock-out phenomena in the Internet. There are lots of AQM schemes, but the most robust scheme is the one, which applies a Proportional Integral Derivative (PID) controller [3]. PID controllers are widely used in the industrial field. They are efficient in providing a robust [4], steadily output. But normally the designed PID controller can not always effectively control systems with changing parameters, because of the fixed PID gains. In order to achieve better QoS [5], the PID controller parameters need to be tuned according to the rapid change of the dynamic network circumstances.

To overcome the drawbacks of static PID controller parameters, some improved methods [6] [7] appear gradually. One of these algorithms can overcome the difficulty of tuning PID parameters under various computer network environments. It utilizes fuzzy control [8] to adjust the parameters of the PID controller, which is called Adaptive Fuzzy PID [9] (AF-PID). The essential idea of AF-PID algorithm is to tune the PID parameters using fuzzy control rules. Fuzzy control is a modern control strategy, which is well-known for dealing with systems under variable loads and uncertain model parameters [10] [11]. Many researchers have focused on AF-PID algorithm, but most of them just proved the performance in a particular simulation environment by Matlab simulation software. In this paper we mainly do two tasks: on the one hand, we operate comparison experiments among Droptail, RED, PID and AF-PID algorithms under seven different scenarios including multi-bottleneck link by NS2 [12]; on the other hand, we build up a network test bed with IPCOP as the router successfully, and embedded with the AF-PID algorithm. Both of these two aspects prove the good performance of AF-PID algorithm.

The rest of this paper is organized as follows. In Section II, we introduce the basic knowledge of Droptail, RED, PID and AF-PID algorithms. NS2 simulation results to validate the algorithm are presented in Section III. Section IV introduces how to apply the proposed algorithm to the test beds. Tests have been done under the test beds to prove the feasibility of AF-PID. Finally, section V concludes the paper.

II. BACKGROUND KNOWLEDGE

As an important complement of end-to-end system congestion control, AQM strategies in Internet routers not only can control the queue length effectively, but also ensure high throughput. Currently, the theoretical methods dealing with congestion in the network layer mainly include Droptail algorithm, RED algorithm,
BLUE algorithm and others based on control theory, such as PID algorithm and AF-PID algorithm. All of these algorithms have great meanings in the development of AQM strategies.

A. Droptail Algorithm

The basic idea of Droptail algorithm [13] is as follows: when packets arrive at the router, they will queue in the buffers, so the buffer capacity should be set large enough. Usually, we set a maximum of queue length, accept packets into the queue until this value, and discard packets arriving later. Droptail algorithm is the most widely used queue management algorithm currently.

B. RED Algorithm

RED algorithm [14] is a superior congestion mechanism than Droptail algorithm. It is divided into two parts: one is calculating the average queue length for estimating the degree of congestion; the other is calculating the probability of packet loss.

RED algorithm calculates the average queue length \( Q_{avg} \) as follows:

\[
Q_{avg} = (1-w_k)Q_{max} + w_kq .
\]

where, \( w_k \) is the weight, \( q \) is the instantaneous sampling queue length. \( w_k \) is a very important parameter, which determines the reaction degree of the router to the changes of input flows. In this way, the growth of the queue length caused by brief congestion will not have serious influence on the average queue length, resulting in filtering out the change of the queue length in a short time, reflecting the long-term changes of congestion as far as possible.

On the other hand, we should calculate the average queue length in order to reflect the status of congestion. RED algorithm has two queue length thresholds: \( T_{min} \) and \( T_{max} \). The relationship between \( Q_{avg} \) and packet loss rate \( p \) is as follows:

- If \( Q_{avg} < T_{min} \), then no packet need to be discarded;
- If \( T_{min} \leq Q_{avg} < T_{max} \), then calculate \( p \) as (2);
- If \( Q_{avg} \geq T_{max} \), then all packets need to be discarded.

Namely:

\[
p = p_{max} \times (Q_{max} - T_{min}) / (T_{max} - T_{min})
\]

where, \( p_{max} \) is the biggest packet loss rate.

C. PID Algorithm

PID control is one of the earliest control strategies because it is simple with good robust performance and higher reliability characteristic. Conventional PID controller is a linear controller. In a PID controller, the input variables are the error signals between expectation and the real value as well as their derivatives, while the outputs are the PID gains. In literature [15], a nonlinear dynamic model for TCP flow control can be approximated as a linear constant system by small-signal linearization about an operating point [16]. The whole TCP control system with PID controller is described in Fig. 1. In the block diagram, \( C(s) \) and \( G(s) \) are the controller and plant, respectively. The error signal \( e \) is defined as: \( e = q - q_0 \). The derivative \( ec \) is defined as: \( ec = e(k) - e(k-1) \), with \( q \) being feedback signal, and \( q_0 \) the desired signal. \( K = (RC)^3/4N \), \( T_i = R \), \( T_z = R^2C/2N \).

So the packet loss rate \( p \) is calculated by using the following equation:

\[
p(k) = K_i e(k) + K_p \sum_{j=0}^{k} e(j) + K_d \left[ e(k) - e(k-1) \right] / T
\]

\[
= K_i e(k) + T_i \sum_{j=0}^{k} e(j) + T_z \left[ e(k) - e(k-1) \right]
\]

(3)

In the discrete-time domain, the PID control law can be expressed as follow:

\[
\Delta p(k) = K_p \left( 1 + T_i + T_z \right) e(k) - \left[ 1 + 2T_i + T_z \right] e(k-1) + T_i e(k-2)
\]

(4)

where, \( T = K_i / K_p \), \( T_i = K_d / K_p \), \( T_z \) stands for the sampling period.

D. AF-PID Algorithm

The fundamental idea of adaptive fuzzy PID controller is to use fuzzy rules to tune PID parameters [17]. The block diagram of AF-PID controller is shown in Fig. 2. \( \Delta K_p \), \( \Delta K_i \) and \( \Delta K_d \) stand for variable quantity of proportional, integral and derivative gains, respectively.

By multiplying scale factor, the real values of \( \Delta K_p \), \( \Delta K_i \) and \( \Delta K_d \) can be produced. \( K_p \), \( K_i \) and \( K_d \) can be adjusted dynamically, they are expressed as follow:

\[
K_p = K_{p0} + \Delta K_p
\]

\[
K_i = K_{i0} + \Delta K_i
\]

(5)

\[
K_d = K_{d0} + \Delta K_d
\]
where, $K_{p0}$, $K_{i0}$ and $K_{d0}$ are the initial values. In this paper, they are designed as follow:

$$K_{p0} = 1.29 \times 10^{-5}; \quad K_{i0} = 2.220 \times 10^{-5}; \quad K_{d0} = 9.50 \times 10^{-6}.$$  

Then we can get the $p(k)$ and $\Delta p(k)$ as the same way with PID algorithm. In conclusion, we can obtain the flow diagram of AF-PID algorithm in Fig. 3.

Performance of AF-PID was compared with Droptail, RED, and PID under different scenarios. From Fig. 5 to Fig. 12, we define that red represents Droptail, green represents RED, blue represents PID and pink represents AF-PID.

A. Different Numbers of TCP Connections

In the simulations, $n$ takes the values of 120 and 1000. The instantaneous queue lengths of four algorithms under different scenarios are recorded in Fig. 5(a) and Fig. 5(b); the throughputs of four algorithms are shown in Fig. 5(c) and Fig. 5(d); Fig. 5(e) and Fig. 5(f) show the jitter of the four algorithms. At last, we can contrast the packet loss rate and delay in Table I. As we can see, the queues of the other three algorithms have different degrees of queue oscillation. Among them, Droptail is completely unable to regulate the queue length around the expected value; PID can’t regulate the queue to the expected level under the heavy load scenario ($n=1000$). In contrast, AF-PID can regulate the queue length nearby the expected length after a short regulating progress under both the light load and the heavy load scenario, which shows that AF-PID is robust to the number of TCP connections. At the same time we can see the throughput, packet loss rate, delay and jitter of AF-PID are all better than the others.

B. Different Delay Time

Fig. 6 presents the instantaneous queue lengths, throughputs and jitters of different algorithms when $d$ takes 20ms and 200ms respectively, and Table II shows the packet loss rate and delay. Obviously, the other three algorithms are unable to adjust the queue lengths at the equilibrium point, especially under the long-delay...
scenario. It is because the probability of empty queue is bigger for Droptail, RED and PID, which easily results in the decrease of link utilization, against the design objective of AQM. On the other hand, the proposed algorithm regulates the queue length at stable state after a short time under the short-delay scenario, and it also adjusts the queue to the desired level under the long-delay scenario, although it has fluctuation at the beginning. Fig. 6(c) and Fig. 6(d) show that the throughput of AF-PID is better than others even in the case of long-delay. Also, we can easily find that AF-PID is less jitter and delay time under both scenarios. As for packet loss rate, AF-PID algorithm performs better under short-delay scenario, while it is slightly larger under long-delay scenario.

for PID and AF-PID algorithms. In this section, different expected queue lengths of PID and AF-PID are set in simulations. The simulation results of several aspects are shown in Fig. 7 and Table III. As it can be seen, PID leads to violently queue oscillation at beginning when the expected is set to 30 packets, and it can't regulate the queue length to 500 packets. However, AF-PID can stabilize the queue to the expected value under two scenarios. In other word, the proposed algorithm can settle down its expected queue length in wide-range which will satisfy the demand of network services. The throughput, jitter, packet loss rate and delay time of AF-PID are all superior to PID in some degrees.

Figure 7. Queue length, Throughput and Jitter for different expected queue length

Table I. Packet Loss Rate and Delay for Different TCP Connections

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Packet loss rate</th>
<th>Delay(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n=120</td>
<td>n=1000</td>
</tr>
<tr>
<td></td>
<td>n=120</td>
<td>n=1000</td>
</tr>
<tr>
<td>DropTail</td>
<td>0.1315</td>
<td>0.1816</td>
</tr>
<tr>
<td></td>
<td>0.1604</td>
<td>0.1930</td>
</tr>
<tr>
<td>RED</td>
<td>0.1088</td>
<td>0.1241</td>
</tr>
<tr>
<td></td>
<td>0.1446</td>
<td>0.1095</td>
</tr>
<tr>
<td>PID</td>
<td>0.0857</td>
<td>0.1349</td>
</tr>
<tr>
<td></td>
<td>0.1189</td>
<td>0.1389</td>
</tr>
<tr>
<td>AF-PID</td>
<td>0.0829</td>
<td>0.1054</td>
</tr>
<tr>
<td></td>
<td>0.1112</td>
<td>0.1123</td>
</tr>
</tbody>
</table>

Table II. Packet Loss Rate and Delay for Different Delay Time

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Packet loss rate</th>
<th>Delay(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>d=200 ms</td>
<td>d=200 ms</td>
</tr>
<tr>
<td></td>
<td>d=200 ms</td>
<td>d=200 ms</td>
</tr>
<tr>
<td>DropTail</td>
<td>0.1791</td>
<td>0.1288</td>
</tr>
<tr>
<td></td>
<td>0.1721</td>
<td>0.3323</td>
</tr>
<tr>
<td>RED</td>
<td>0.1167</td>
<td>0.1031</td>
</tr>
<tr>
<td></td>
<td>0.1295</td>
<td>0.3017</td>
</tr>
<tr>
<td>PID</td>
<td>0.0981</td>
<td>0.0836</td>
</tr>
<tr>
<td></td>
<td>0.1223</td>
<td>0.2931</td>
</tr>
<tr>
<td>AF-PID</td>
<td>0.0810</td>
<td>0.0976</td>
</tr>
<tr>
<td></td>
<td>0.1214</td>
<td>0.2879</td>
</tr>
</tbody>
</table>

C. Different Expected Queue Length

Expected queue length qref is an important parameter

Table III. Packet Loss Rate and Delay for Different Expected Queue Length

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Packet loss rate</th>
<th>Delay(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>qref=30 (packet)</td>
<td>qref=500 (packet)</td>
</tr>
<tr>
<td></td>
<td>qref=30 (packet)</td>
<td>qref=500 (packet)</td>
</tr>
<tr>
<td>PID</td>
<td>0.1492</td>
<td>0.0323</td>
</tr>
<tr>
<td></td>
<td>0.0597</td>
<td>0.1635</td>
</tr>
<tr>
<td>AF-PID</td>
<td>0.1059</td>
<td>0.0306</td>
</tr>
<tr>
<td></td>
<td>0.0426</td>
<td>0.1551</td>
</tr>
</tbody>
</table>

D. Different Bottleneck Link Capacity

In order to test the performance of AF-PID under different bottleneck link capacities, two scenarios have been set. In the simulations, the capacity C is set to 2Mbps and 40Mbps. Fig. 8 and Table IV show the simulation results under two scenarios respectively. As we can see, the queue oscillation of the other three algorithms under both scenarios is much bigger than AF-PID, especially Droptail and RED, barely stabilize the queue to the expected value. In contrast, AF-PID can regulate the queue length to the target level under two
different capacities, which shows the robustness to the bottleneck link capacity. Meanwhile, it can be seen that not only throughput, but also jitter, packet loss rate and delay time are all better than the other three algorithms, though it is not so obvious as the queue length.

![Figure 8. Queue length, Throughput and Jitter for different bottleneck link capacity](image)

**TABLE IV.** PACKET LOSS RATE AND DELAY FOR DIFFERENT BOTTLENECK LINK CAPACITY

<table>
<thead>
<tr>
<th>Property</th>
<th>Algorithm</th>
<th>C=2 Mbps</th>
<th>C=40 Mbps</th>
<th>C=2 Mbps</th>
<th>C=40 Mbps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DropTail</td>
<td>0.1169</td>
<td>1.0539</td>
<td>1.0610</td>
<td>0.0754</td>
</tr>
<tr>
<td></td>
<td>RED</td>
<td>0.1087</td>
<td>0.0882</td>
<td>0.7102</td>
<td>0.2228</td>
</tr>
<tr>
<td></td>
<td>PID</td>
<td>0.0857</td>
<td>0.0891</td>
<td>0.6951</td>
<td>0.0622</td>
</tr>
<tr>
<td></td>
<td>AF-PID</td>
<td>0.0729</td>
<td>0.0753</td>
<td>0.6265</td>
<td>0.0395</td>
</tr>
</tbody>
</table>

**TABLE V.** PACKET LOSS RATE AND DELAY WITH UDP CONNECTIONS

<table>
<thead>
<tr>
<th>Property</th>
<th>Algorithm</th>
<th>udp=10</th>
<th>udp=50</th>
<th>udp=10</th>
<th>udp=50</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DropTail</td>
<td>0.1791</td>
<td>0.2288</td>
<td>0.1914</td>
<td>0.1345</td>
</tr>
<tr>
<td></td>
<td>RED</td>
<td>0.1864</td>
<td>0.1807</td>
<td>0.1173</td>
<td>0.1447</td>
</tr>
<tr>
<td></td>
<td>PID</td>
<td>0.1761</td>
<td>0.1912</td>
<td>0.1302</td>
<td>0.1843</td>
</tr>
<tr>
<td></td>
<td>AF-PID</td>
<td>0.1709</td>
<td>0.1754</td>
<td>0.1098</td>
<td>0.1097</td>
</tr>
</tbody>
</table>

**E. Mixed with UDP Connections**

To examine the performance of AF-PID when there are non-responsive flows in the network, we set two scenarios. There are 190 TCP connections and 10 UDP connections in the first scenario, and 150 TCP connections and 50 UDP connections in the second. Transmission rate of UDP is 1Mbps and packet size is 1000 bytes. The evaluation results are shown in Fig. 9 and Table V. As it can be seen, Droptail, RED and PID algorithms have lost the regulating capacity under the two scenarios. AF-PID maintains the queue at the equilibrium point all the time under two scenarios, and also does well in the other four aspects, which prove that the proposed algorithm has robustness to non-responsive flows.

![Figure 9. Queue length, Throughput and Jitter with UDP connections](image)

**Figure 9. Queue length, Throughput and Jitter with UDP connections**

![Figure 10. Queue length, Throughput and Jitter for dynamic TCP connections](image)

**Figure 10. Queue length, Throughput and Jitter for dynamic TCP connections**
F. Dynamic TCP Connections

Finally, we investigate the robustness of AF-PID under the dynamic TCP connections situation. In the first scenario, \( n \) is set to 200 initially and increases to 300 at 20s. In the second scenario, \( n \) is set to 300 and decreases to 200 at 20s. Fig. 10 and Table VI show the simulation results of four algorithms under different scenarios. Apparently, the queue lengths of the other three algorithms have large fluctuation when the TCP connections are changed while AF-PID not. The adaptive turning scheme of AF-PID reduces the fluctuation when the TCP connections are changed while AF-PID not. The adaptive turning scheme of AF-PID reduces the fluctuation effectively and achieves the desired level. Although AF-PID is not so outstanding at throughput control, its performance on time delay and jitter is obviously superior to other algorithms. Therefore we can conclude that the AF-PID possesses the adaptability under the dynamic network environment effectively.

G. Performance Comparision of the Four Algorithms in the Case of Multi-Bottleneck Link

In order to verify the performance of the AF-PID algorithm in multi-bottleneck link environment, we use the simulation topology shown in Fig. 11. The bandwidths and the delay times of every link are marked in Figure11. The buffer size for each link is 600 packets. The expected queue length of PID and AF-PID in every scenario is 300 packets. Compared with the single bottleneck link, multi-bottleneck links increase to 6 routers and two sets of TCP cross-data flows. There are two bottleneck links in the topology: one lies between \( R_1 \) and \( R_2 \), the other lies between \( R_3 \) and \( R_4 \) respectively. We use Droptail, RED, PID and AF-PID algorithms in the bottleneck link routers \( R_1 \) and \( R_2 \) respectively. The simulation time is 50 seconds.

Table VI. Packet Loss Rate and Delay for Dynamic TCP Connections

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Packet loss rate</th>
<th>Delay(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dynamic 200-300</td>
<td>dynamic 300-200</td>
</tr>
<tr>
<td>DropTail</td>
<td>0.1337</td>
<td>0.2689</td>
</tr>
<tr>
<td>RED</td>
<td>0.1622</td>
<td>0.1490</td>
</tr>
<tr>
<td>PID</td>
<td>0.1513</td>
<td>0.1247</td>
</tr>
<tr>
<td>AF-PID</td>
<td>0.1229</td>
<td>0.1169</td>
</tr>
</tbody>
</table>

Figure 11. Simulation topology of multi-bottleneck link

The simulation results are shown in Fig. 12 and Table VII. AF-PID is able to adjust the queue to the desired level, although it has fluctuation at the beginning, and the packet loss rate and throughput have a slight improvement. It turned out that the AF-PID algorithm is better than the other algorithms comprehensively.

IV. APPLYING AF-PID TO TEST BED

A. Structure of Test Bed

The topology of test bed is shown in Fig. 13, it consists of sending module, routing module and receiving module. Sending module is composed of four HP dx2700MT computers (CPU: E2180, Petium dual core 2.0; Memory: 512MB*2, DDR2667; Main Frequency: 2.0GHz; Hard-disk: 160GB, SATA; Network Card: Realtek RTL8168C; Operating System: WinXP PRO SP2). The routing module consists of IPCOP software router \( R_1 \) and H3 AR18-63-1 router \( R_2 \) (Flash Memory: 16MB, DRAM Memory: 256MB, Processor: Mips processor 800MHz). Receiving module is composed of one HP dx2700MT computer.

Figure 12. Queue length, Throughput and Jitter for multi-bottleneck link

Figure 13. The topology of real LAN
B. Technique of Embedding the AF-PID into IPCOP Routers

IPCOP is an operating system which possesses routing function and high stability, and it can provide user with open source codes and interface of embedding new congestion algorithms. So, it is convenient to investigate network congestion control by using IPCOP.

The source codes of IPCOP consist of ipcop-sources.gz, ipcop-othersrc.tar.tar and ipcop-othersrc.tar.bz2.md5. The main codes realizing the congestion control lie in linux and iproute. If users want to add new algorithm into IPCOP, the above two files should be revised. In this paper, AF-PID is embedded, and the steps are described as Fig. 14.

After revising and recompiling ipcop1.4.16, an image file of IPCOP is obtained, and it can be installed on PC as a router.

C. Experiments on Test Bed

AF-PID is embedded in R1. The parameters of AF-PID are the same as the NS simulations in section III. Every sender sends 30 files with 10.7MB size using TCP protocol. The sizes of all buffers are 600 packets and the expected queue length is 300 packets.

![Figure 14. The flow chart of embedding new algorithm](image)

**Figure 14.** The flow chart of embedding new algorithm

Fig. 15 shows the queue length of AF-PID under the real network. As it can be seen, the new algorithm can stabilize queue length around the expected length after a short regulating time (about 4s), which shows the effectiveness and robustness of the AF-PID. So, we can conclude that the AF-PID is feasible under the real network circumstance.

![Figure 15. The queue length of AF-PID in the real LAN](image)

**Figure 15.** The queue length of AF-PID in the real LAN

V. Conclusions

The traditional AQM algorithms are difficult to obtain a good control effect due to the dynamic behavior of the network traffic, especially under the heavy-load and long-delay time scenarios. However, AF-PID doesn’t depend on accurate model by introducing fuzzy reasoning and adding detailed fuzzy rules. The performance of AF-PID is evaluated in NS2 simulations and real network circumstances. The simulations are easier to operate than in Matlab because the codes are written by Otcl language in NS2, which is convenient to change scenarios. More important, the simulation results for communication network are closer to the actual situation in NS2. Simulation results show that AF-PID always has better performance than Droptail, RED and PID algorithms, including queue length, throughput, packet loss rate, delay and jitter under different network scenarios. Besides by the way we find to embed the AF-PID algorithm to the real network routers, we find that AF-PID algorithm still keep its performance as in the simulation experiments, and the implementation is easy to realize, resource consumption is in the normal range. Thus we can conclude that AF-PID is an effective algorithm and provides higher Quality of Service in network.

ACKNOWLEDGMENT

This work was supported by the Key Scientific and Technological Project of Jilin Province (No. 201115030), and Natural Science Foundation of Jilin Province (No. 201115030).

REFERENCES


**Xiaoping Yang** received the B.E. and M.E. degree in Detection Technology and Instrument and Communication and Electronic System from Jilin University of Technology, Changchun, China, in 1985 and 1988, respectively, and Ph.D. degree in Control Theory and Control Engineering, Jilin University, Changchun, China in 2007. She is a professor at College of Communication Engineering, Jilin University, China. She has been committed to teaching and research including communication, signal and information processing for years. Her current research interest includes optimization control of data transmission and congestion control in communication network.

**Chunfeng Song** is currently a postgraduate and working towards her M.E. degree in Jilin University, China. Her current research interest includes active queue management algorithm.

**Jiwei Sun** is currently a postgraduate and working towards his M.E. degree in Jilin University, China. His current research interest includes simulation implementation of AQM algorithm.