International Journal of Networking and Computing – www.ijnc.org
ISSN 2185-2839 (print) ISSN 2185-2847 (online)
Volume 7, Number 2, pages 295–317, July 2017

A Minimum Contention Window Control Method for Lowest Priority Based on Collision History of Wireless LAN

Tomoki Hanzawa
Graduate School of Systems and Information Engineering,
University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, Japan

and

Shigetomo Kimura
Faculty of Engineering, Information and Systems,
University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, Japan

Received: February 14, 2017
Revised: May 6, 2017
Accepted: May 30, 2017
Communicated by Satoshi Ohzahata

Abstract

Because of the widespread adoption of mobile devices, many applications have provided support for wireless LAN (WLAN). Under these circumstances, one of the important issues is to provide good quality of service (QoS) in WLAN. For this purpose, Dhurandher et al. improved the distributed coordination function (DCF). In this method, the contention window (CW) is divided into multiple ranges. Each range is independent of all other ranges and is assigned to a different priority. Although the highest-priority throughput increased using this method, throughput for the other priorities decreased significantly. To overcome this problem, this paper proposes a minimum contention window control method for two (high and low) priorities. In the method, all nodes are assumed to use real-time applications or data transmission. The former real-time frames are high priority and are sent by UDP. The latter data frames are low priority and are sent by TCP. The purpose of the proposed method is not only to provide good QoS for the highest priority but also to prevent deterioration in the QoS for other priorities in WLAN. For this purpose, the proposed method keeps the CW for the high priority at a low value and controls the CW for the low priority based on the collision history. Finally, the network simulations demonstrated that the proposed method reduces the decrease in the average total throughput of the low priority frames as well as reducing the packet drop rate of both priorities, compared with those for the DCF and Dhurandher's method. From a simulation scenario where there are only low priority flows in wider bandwidths, all methods give almost same average total throughput and packet drop rate, but the results also suggest that the CW range in the proposed method should be reduced to improve the average total throughput, when no congestion occurs.

Keywords: Minimum Contention Window Control Method, Collision History, DCF, Wireless LAN

1 Introduction

Recently, wireless LAN (WLAN) has become a major communication method as WLAN devices are cheaper and have improved performance. WLAN is usually used within the home and in offices, while public WLAN services are also popular. As the use of mobile devices such as smartphones and tablets has become widespread, various Internet services are also provided via WLAN.

Under these circumstances, one of the most important issues is to provide good quality of service (QoS) in WLAN. For example, for real-time applications like voice or video streaming, if the packet loss and/or jitter increases, then QoS rapidly decreases. Therefore, the packets from such real-time applications should be transmitted with higher priority than data packets. However, WLAN that is based on IEEE 802.11 cannot guarantee QoS because it processes these packets in a simple first-in first-out order.

Various methods are proposed to guarantee QoS in WLAN. The IEEE 802.11 working group proposed IEEE 802.11e and IEEE 802.11aa for this purpose. Under these standards, various methods are proposed to guarantee QoS [1], [2].

Because these standards are optional, most of the devices in the consumer market do not support them. Thus, it is difficult to adopt the proposed methods on a typical WLAN network. Furthermore, in [2], it is reported that throughput is not sufficient when IEEE 802.11e and IEEE 802.11ac are used in combination. Therefore, it is impossible for a public WLAN to guarantee QoS since many unspecified users exist.

Many methods have also been proposed for WLAN devices that do not use IEEE 802.11e and/or IEEE 802.11aa to guarantee QoS. In [3], traffic is categorized into two priority classes. When a low priority frame is sent, this method generates a random number. If this number exceeds a threshold, the frame is allowed to attempt sending. In [4], in an environment where data packets and real-time packets are queued together for sending, before a packet is inserted into the queue on a WLAN access point, it may be discarded based on a probability calculated using the queue length. In [5], Dhurandher et al. improved the distributed coordination function (DCF), which is the mandatory media access control in WLAN. In this method, the contention window (CW) is divided into multiple ranges. Each range is independent of all other ranges and is assigned to a different priority. Although the throughput of the highest priority frames increased using this method, throughput for the other priorities decreased significantly. To overcome this problem, this paper proposes a minimum contention window control method for two (high and low) priorities. In the method, all nodes are assumed to use real-time applications, such as voice and video streaming, or data transmission, such as FTP, HTTP, etc. The former real-time frames are high priority and are sent by UDP. The latter data frames are low priority and are sent by TCP. The purpose of the proposed method is not only to provide good QoS for the high priority frames, but also to prevent a decrease in the QoS for the low priority frames. For this purpose, the proposed method keeps the CW for the high priority at a low value, and controls the CW for the low priority based on the collision history.

It should be noted that priority controls for QoS can also be inserted after the CW. However, these controls are categorized by packet scheduling methods such as WFQ (Weighted Fair Queuing) [6], [7]. Since the controls can be provided independently or cooperatively with the proposed method, they are beyond of the scope of this paper.

Finally, in the network simulations, the average total throughput and packet drop rate for TCP and UDP flows were observed, and the results showed that the proposed method reduces the decrease in the average total throughput of the low priority frames and the packet drop rate of both priorities, compared with those where the DCF and Dhurandher's method are used. From a simulation scenario where there are only low priority flows in wider bandwidths, all methods gave almost the same average total throughput and packet drop rate, but the results also suggest that the CW range in the proposed method should be reduced to improve the average total throughput, when no congestion occurs.

The rest of this paper is organized as follows. Section 2 introduces the DCF and Dhurandher's method. Section 3 proposes a minimum contention window control method for the lowest priority based on the collision history. Section 4 presents the communication experiments. Section 5 con-

cludes the paper and discusses the case where the proposed method and unspecified users using other access control methods are mixed. This section also describes future work.

2 Contention Window Control Method for QoS

This section introduces the distributed coordination function (DCF) and Dhurandher's method [5].

2.1 Distributed Coordination Function (DCF)

The distributed coordination function (DCF) is the mandatory distributed channel access method defined in IEEE 802.11. It is also referred to as carrier sense multiple access with collision avoidance (CSMA/CA). Suppose the infrastructure network has an access point (AP) and nodes A and B. As shown in Figure 1, when all nodes including the AP try to send a frame, each node senses a carrier on the channel. If the channel is idle for the duration of the DCF interframe space (DIFS), each node generates a random number and starts the backoff timer, the expiration time of which is the slot time multiplied by the random number. All nodes decrease their own backoff time. In Figure 1, node A transmits a data frame because its backoff timer reaches zero first. The other two nodes detect the carrier and then stop decreasing their own backoff times.

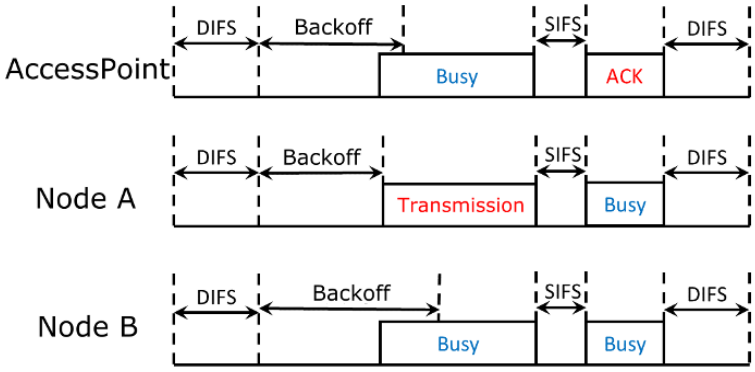

Figure 1: Media Access Control by CSMA/CA.

When the data frame is received successfully, the receiver waits for the duration of the short interframe space (SIFS) and then transmits an acknowledgment (ACK) control frame. Only when node A receives the ACK control frame does the transmission succeed. When the ACK control frame is lost, or more than two data frames are transmitted with node A because their backoff timers reach zero simultaneously, then a collision occurs and both frame transmissions will fail. After the channel is idle for the next DIFS period, node A decreases the new transmission time by generating a random number as follows to retransmit the frame, while the AP and node B also restart decreasing their own backoff times.

In the above procedures, a random number is generated between 0 and CW. The CW ranges from the minimum CW (CW_{min}) to the maximum CW (CW_{max}) and is represented by the following equation:

$$CW = \min((CW_{min} + 1) \times 2^n - 1, CW_{max}) \quad (1)$$

where n is the number of successive collisions. When a node first tries to transmit a frame, the random number is generated from 0 to CW_{min}. For example, CW_{min} is 7 in Figure 2. The CW range of the random number for the first transmission is from 0 to 7 as in the top row in the figure. When the first transmission fails, the CW range of the random number doubles as shown in the middle row in Figure 2. If the second transmission also fails, the range doubles as shown in the bottom row in the figure. After the frame is transmitted correctly, the sender resets CW to CW_{min}.

In this way, if the transmission continues to fail, then CW is exponentially increased to decrease the probability of collisions.

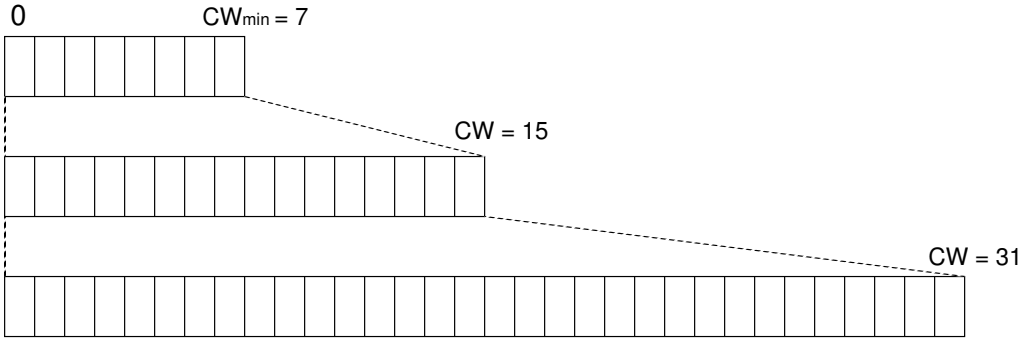

Figure 2: Expansion of CW range by collisions.

The above algorithm is called binary exponential backoff (BEB) and has also been adopted for the Ethernet. Since the backoff algorithm has an impact on throughput in WLAN, several backoff algorithms have been proposed. For example, the multiplicative increase and linear decrease (MILD) algorithm slows down the CW change, and smart exponential threshold linear (SETL) controls the CW based on a threshold and the transmission success times [8]. However, the more nodes that try to send a frame, the more often collisions tend to occur. This causes longer backoff times and decreases throughput in WLAN [9]. Moreover, the current DCF is designed such that all nodes are equally likely to be able to transmit their frame. This fact also causes unequal transmission balance between uplink and downlink, and thus each application cannot guarantee QoS [10], [11].

2.2 Dhurandher's method

In [5], a contention window control method is proposed to guarantee QoS. Although the DCF generates a random number regardless of the priority of a frame, in this method, the CW range is divided into non-overlapped independent priority levels. For example, the CW range is divided into four priority levels as shown in Figure 3. When a node sends a frame at the highest priority, it generates a random number within the range of Priority 0. When a node sends at the lowest priority, the number is generated within Priority 3. When a collision occurs, the CW range doubles, but the priority levels do not overlap with each other, as shown in the middle and bottom rows in Figure 3. By generating random numbers from different ranges based on the priority, specific frames at higher priorities can be transmitted rapidly.

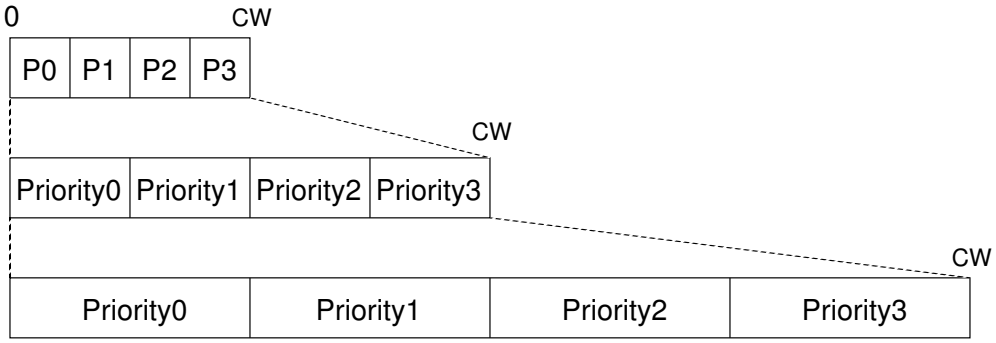

Figure 3: CW range divided by priority levels.

Using this method, throughput at Priority 0 is improved. However, since Priority n must always wait for the range from Priority 0 to Priority $n-1$, the throughput for those priorities is decreased

significantly compared to that of Priority 0.

To solve this problem, the next section improves Dhurandher's method for two priority levels and proposes a method to define the CW range at each priority. This method attempts not only to increase high priority throughput, but also to prevent a decrease in low priority throughput.

3 Minimum Contention Window Control Method for Lowest Priority Based Collision History

This section assumes two (high and low) priorities and proposes a minimum contention window control method for the low priority based on the collision history. In the method, all nodes are assumed to use real-time applications, such as voice and video streaming, or data communication, such as FTP, HTTP, etc. The former real-time frames are high priority and are sent by UDP. The latter data frames are low priority and are sent by TCP.

3.1 Contention window range at first transmission

Since Dhurandher's method divides the CW range into non-overlapped independent priority levels, throughput except at the highest priority level is greatly decreased. Therefore, in the proposed method, the low priority CW range includes that of the high priority, as shown in Figure 4. For the first transmission, the latter range is the lower half of the former range.

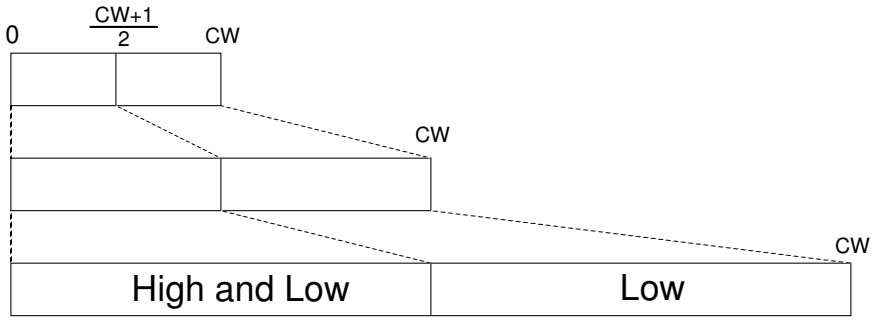

Figure 4: Expansion of CW range by proposed method.

When a collision occurs, each CW range doubles as in the DCF. Then, the CW of each priority level is defined as:

$$\mathrm{CW} = \begin{cases} \min\left(\dfrac{(\mathrm{CW_{min}}+1)\times 2^n}{2} - 1, \dfrac{\mathrm{CW_{max}}+1}{2} - 1\right) & \text{(High priority)} \\ \min\left((\mathrm{CW_{min}}+1)\times 2^n - 1, \mathrm{CW_{max}}\right) & \text{(Low Priority)} \end{cases} \tag{2}$$

where $\mathrm{CW_{min}}$ and $\mathrm{CW_{max}}$ are the initial and maximum values of the CW defined in IEEE 802.11 and n is the number of successive collisions. From the above equation, a node generates a random number from 0 to $\frac{(\mathrm{CW_{min}}+1)\times 2^n}{2} - 1$ to transmit a high priority frame. If a collision occurs, the CW range doubles until the range reaches $\frac{(\mathrm{CW_{max}}+1)\times 2^n}{2} - 1$. When the node transmits a low priority frame, it generates a random number from 0 to $(\mathrm{CW_{min}}+1)\times 2^n - 1$ as defined in IEEE 802.11. If collisions occur, the CW doubles up to $\mathrm{CW_{max}}$. For the high priority, the CW range is always half of that of the low priority, i.e., that of the DCF, for the same n, until the first transmission succeeds. Therefore, the probability of transmitting a high priority frame is greater than for a low priority frame. In addition, since the low priority CW range is included in the high range, a low priority frame can be transmitted more easily than in Dhurandher's method.

3.2 Contention window range after transmission succeeds

As shown in Figure 5, the proposed method also changes the CW range after a node succeeds in transmitting the frame. In Dhurandher's method, after transmission succeeds, the CW range is reset to the initial range of each priority. In the proposed method, the high priority CW range is also reset from 0 to $\frac{CW_{min}+1}{2} - 1$, but the low priority range is set from 0 to half of the current CW. If the half-CW value is smaller than CW_{min}, then the value becomes CW_{min}. The range is based on the collision history because the range is too wide to reduce the probability of transmitting low priority frames when the node experiences a high collision rate for these frames. When the node does not send a low priority frame for the period of CW × Slot_time, the proposed method reduces the current CW range by half. Using the above control, low priority frames do not significantly prevent high priority frames from transmitting continuously.

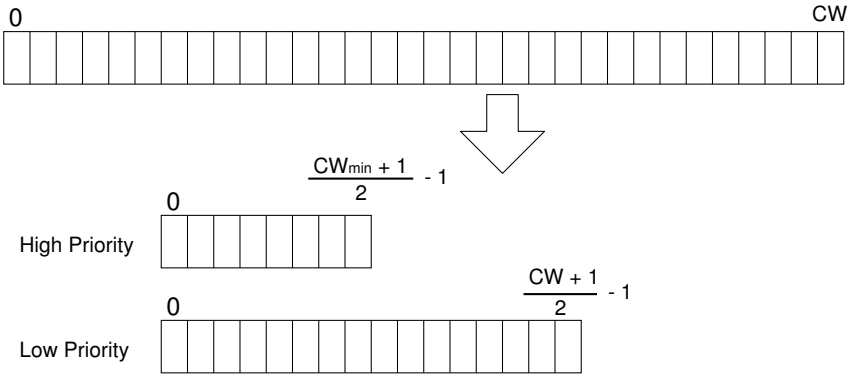

Figure 5: CW range after transmission succeeds.

4 Simulation Experiments

This section describes the evaluation of the proposed method, which uses network simulator 3 to implement the proposed method and executes the network simulation experiments to compare the results of the proposed method with the DCF and Dhurandher's method.

In the following, there are three simulation experiments referred to as Scenarios 1, 2 and 3. In each scenario, all nodes transmit frames to the same WLAN access point (AP). In Scenarios 1 and 2, since the AP utilizes a narrow bandwidth, the performance in the congested situation is estimated. Scenario 3 uses an AP with a wider bandwidth to evaluate the performance in a lightly congested network.

4.1 Simulation environment

Table 1 lists the experimental conditions that are common parameters for Scenarios 1 and 2. The network topologies in Scenarios 1 and 2 are shown in Figure 6 and Figure 7, respectively. TCP flows always transmit data by FTP. UDP flows are assumed to be bidirectional voice communications between the access point (AP) and nodes. The UDP packet size is 320 bytes and the flow is a constant bit rate (CBR) of 64 Kbps. The simulation time is 100 seconds.

In Scenario 1, five bidirectional UDP flows and five TCP downlink flows from the AP to the nodes are fixed as shown in Figure 6. The number of TCP uplink flows from the nodes to the AP is modified from 1 to 10 to show the influence from the low priority uplink flows.

In Scenario 2, five TCP downlink and five TCP uplink flows are fixed as shown in Figure 7. The number of UDP flows is modified from 1 to 10. Since the UDP bidirectional flows with high priority are increased, the collision history in TCP flows with low priority may be affected. The purpose of this scenario is to check the impact of the collision history on the proposed method.

Table 1: Experimental Conditions for Scenarios 1 and 2.

Simulator	Network Simulator 3
Wireless LAN Standard	IEEE 802.11b
Minimum CW	31
Maximum CW	1023
Data Rate	11 Mbps
Maximum Retransmission Times	7 times
Simulation Time	100 [sec]
Simulation Count	10 counts
Node's Queue type	Single Drop-Tail Queue
Node's Queue length	100 packets
UDP Traffic Pattern	Real Time Application (VoIP)
UDP Packet Size	320 Bytes
UDP Bit Rate	64 Kbps (CBR)
TCP Traffic Pattern	FTP
TCP Maximum Segment Size	1460 Bytes

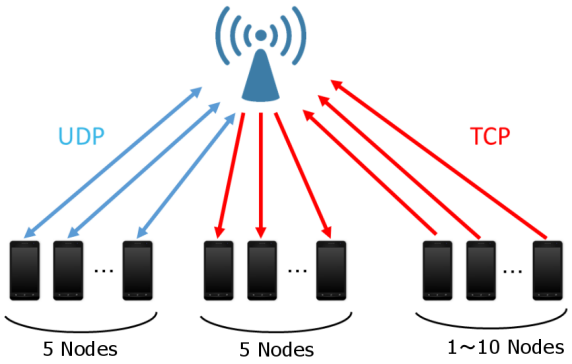

Figure 6: Topology in Scenario 1.

In Scenarios 1 and 2, IEEE 802.11b is used because it is mainly supported in network simulator 3 and it is easy to generate a congested situation from the narrow bandwidth. The simulation results can be applied to the current wireless LAN standards such as IEEE 802.11ac. IEEE 802.11ac provides higher bit rates and less overhead such as a shorter preamble, SIFS and DIFS durations are shorter than those of IEEE 802.11b. Therefore, if IEEE 802.11ac is used in Scenarios 1 and 2, the average total throughput will increase. However, IEEE 802.11ac also adopts the DCF, the transmission mechanism of which is the same as IEEE 802.11b. The single-user MIMO (Multi-Input Multi-Output) will not affect the results since all antennas of the AP and nodes use the same frequency to transmit frames with the result that the collision probability is not improved by MIMO. Although the multi-user MIMO may reduce the collision probability when the number of nodes is less than the number of antennas, these reductions have less effect since there are 10 or more nodes in Scenarios 1 and 2, which is greater than the maximum number of antennas in IEEE 802.11ac. Thus, if TCP flows also increase until the bandwidth is consumed, then results similar to those shown in the next section will be obtained. However, we should also consider cases where TCP flows cannot consume the bandwidth. That is why the next scenario needs to be examined.

In Scenarios 1 and 2, the network is heavily congested. However, the proposed method may also give a wider CW range to lower priority flows, even when the network congestion is light. Scenario 3 confirms whether or not the proposed method has any influence on the lower priority flows under the conditions listed in Table 2. First, to just show the influence, no higher priority flows are transmitted

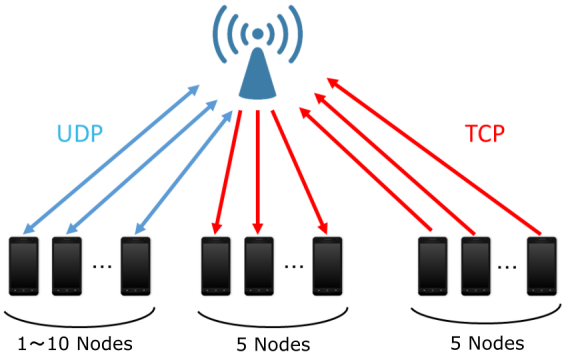

Figure 7: Topology in Scenario 2.

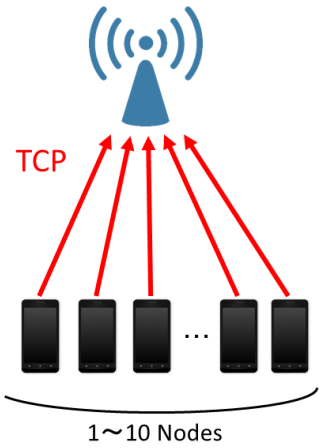

Figure 8: Topology in Scenario 3 (before).

and IEEE 802.11g is used to provide wide bandwidth. As shown in Figure 8, only 1 to 10 TCP uplink flows are transferred to show the influence on the low priority flows. Then, as shown in Figure 9, five bidirectional UDP flows are added to evaluate the performance of high priority flows under light congestion.

Under the experimental conditions, the simulations for each scenario are executed 10 times to measure the average and 95% confidence interval of the total throughput and packet loss rate.

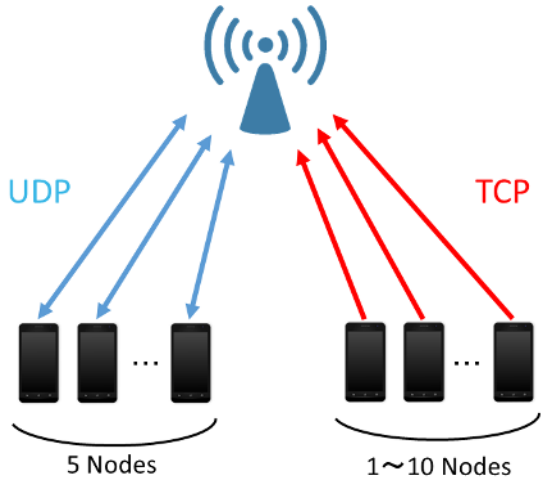

Figure 9: Topology in Scenario 3 (after).

Table 2: Experimental Conditions for Scenario 3.

Simulator	Network Simulator 3
Wireless LAN Standard	IEEE 802.11g
Minimum CW	15
Maximum CW	1023
Data Rate	54 Mbps
Maximum Retransmission Times	7 times
Simulation Time	100 [sec]
Simulation Count	10 counts
Node's Queue type	Single Drop-Tail Queue
Node's Queue length	100 packets
TCP Traffic Pattern	FTP
TCP Maximum Segment Size	1460 Bytes

4.2 Simulation results of Scenario 1

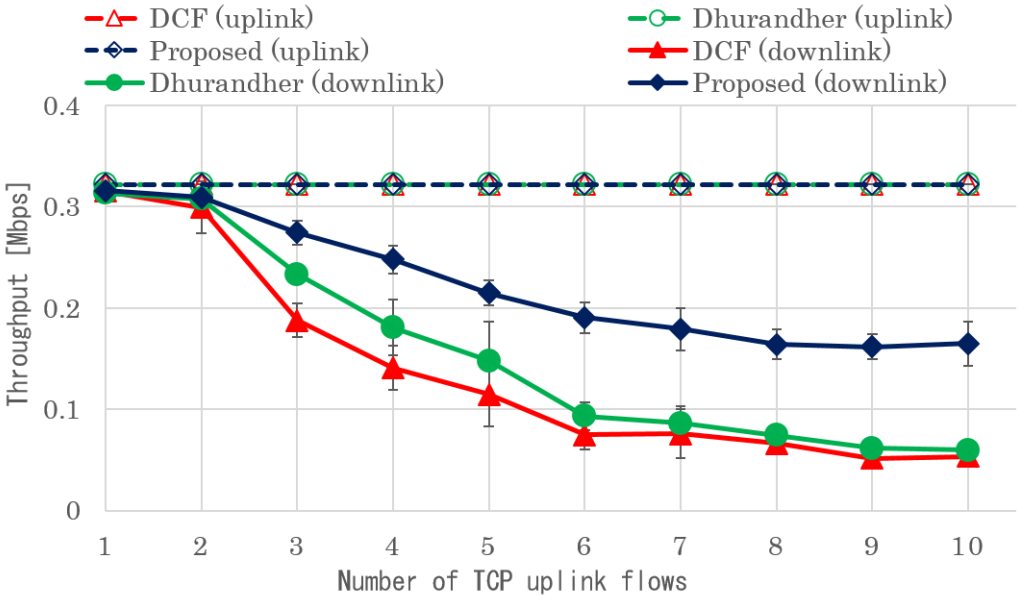

Figure 10: Average total UDP throughputs in Scenario 1.

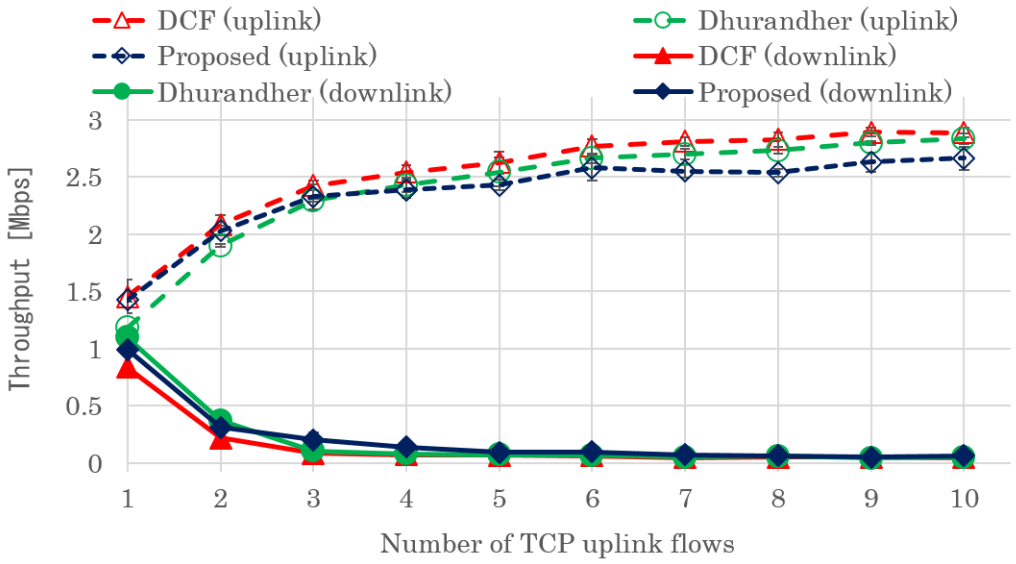

Figure 11: Average total TCP throughputs in Scenario 1.

Figure 10 and Figure 11 respectively show the average total UDP and TCP throughputs in Scenario 1. In Figure 10, even when the TCP uplink flows increased, the average total throughputs of the UDP uplink flows for all methods were about 0.32 Mbps (= 64Kbps × 5) because the UDP bit rate is low and the nodes that transmit UDP datagrams can generate a random number for transmitting independently.

However, the throughputs of the UDP downlink flows decreased in all methods because only the AP can generate a random number to transmit UDP datagrams and then decrease the opportunity to transmit frames when the TCP uplink flows increased. However, the proposed method improved the average total throughput from 0.001 to 0.116 Mbps compared to the DCF and Dhurandher's method. Since the AP queued both high and low priority frames in the same queue, if the AP takes a long time to transmit low priority frames, then it also takes a long time to transmit high priority frames. In Dhurandher's method, when low priority frames are transmitted, they must wait for the high priority CW range. Meanwhile, in the proposed method, because the low priority CW range includes that of high priority, a low priority frame can be transmitted sooner than in Dhurandher's method.

In Figure 11, when the TCP uplink flows increased, the average total TCP throughputs of uplink flows in all methods increased. The results of Dhurandher's method were from 0.05 to 0.27 Mbps lower than those of the DCF. The proposed method improved throughput from 0.03 to 0.24 Mbps when the number of TCP uplink flows were from 1 to 3, since Dhurandher's method has a longer delay before transmission than the proposed method under this condition. However, when the number of TCP uplink flows was greater than 3, the average total throughput of the proposed method decreased from 0.03 to 0.24 Mbps compared with the other two methods. However, the decrement contributed to the increase in the average total throughput of the UDP flows in the proposed method.

On the other hand, the average total throughputs of TCP downlink flows decreased when the number of TCP uplink flows increased since the TCP congestion control may almost stop the communications under a higher packet loss ratio. But the proposed method improved throughput from 0.001 to 0.117 Mbps compared with the other two methods. Only the AP can transmit frames to the downlink, and in the AP queue, high and low priority frames are mixed. Since the proposed method can transmit low priority frames sooner than Dhurandher's method, the proposed method can transmit frames from the queue faster than the other methods.

Figure 12 and Figure 13 show the UDP and TCP packet drop rates. In Figure 12, the rates for uplink flows for all methods were from 0 to 0.033%. However, the rates for downlink flows increased

very markedly when the number of TCP uplink flows increased. Furthermore, when the number of TCP uplink flows was 3 or more, the proposed method improved from 25.7 to 33.6% over the DCF, and from 11.8 to 33.6% over Dhurandher's method.

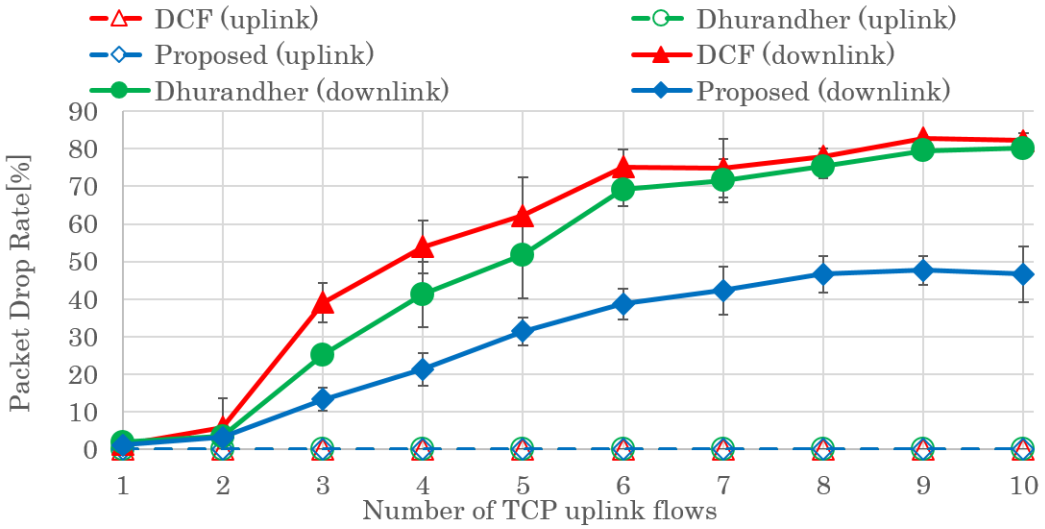

Figure 12: Average UDP packet drop rates in Scenario 1.

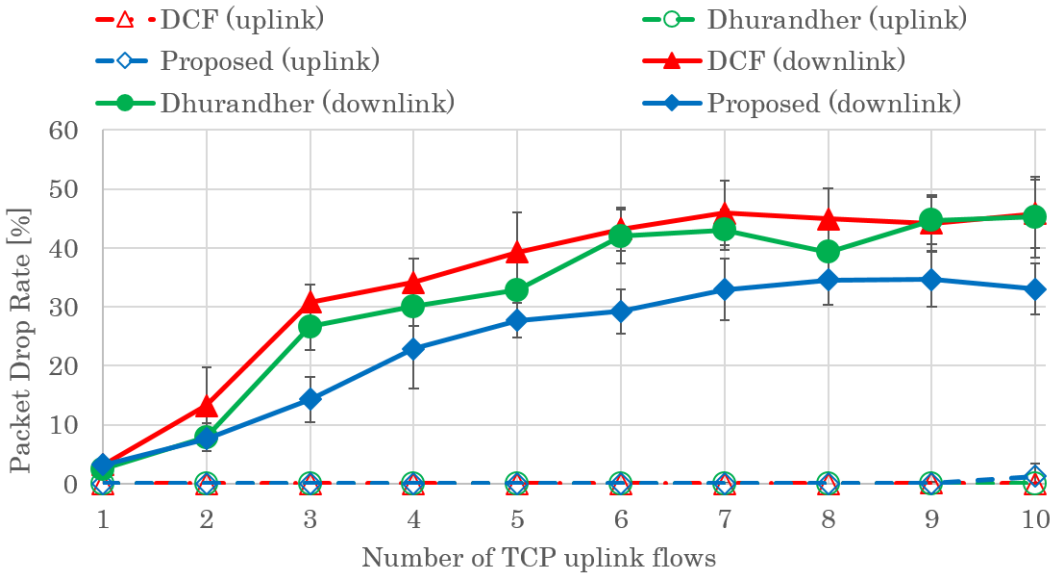

Figure 13: Average TCP packet drop rates in Scenario 1.

In Figure 12, all methods had almost the same TCP uplink packet drop rate, from 0 to 1.25%. On the other hand, the proposed method improved the TCP downlink packet drop rate from 0.3 to 16.5% over the other two methods, when the number of TCP uplink flows was 2 or more.

To evaluate the QoS of real-time traffic, Figure 14 and Figure 15 show the average UDP packet delay and jitter, respectively. Table 3 also shows the average fairness index [12] for UDP flows.

In Figure 14, the average delay for uplink flows in all the methods was almost 0.0 ms. Although the average delay for downlink flows increased significantly when the number of TCP uplink flows

increased, the proposed method improved from 170.3 to 683.0 ms over the DCF, and from 107.0 to 594.0 ms over Dhurandher's method.

In Figure 15, the average jitter in uplink flows for all methods was from 4.8 to 13.6 ms. While the average jitter in downlink flows increased from 21.5 to 59.8 ms for the DCF and from 25.3 to 62.2 ms for Dhurandher's method when the number of TCP uplink flows was more than 4, that of the proposed method was lower than 29.0ms even when the number of TCP uplink flows was 10.

In Table 3, the average fairness index was always 1.00 and thus fairness for each UDP uplink flow in all methods was maintained. However, the average fairness index for the DCF and Dhurandher's method for each UDP downlink flow shifted from 0.76 to 0.54 and from 0.83 to 0.55, respectively, when the number of TCP uplink flows was more than 3. Since that of the proposed method was 0.95 even when the number of TCP uplink flows was 10, fairness for each UDP uplink flow was maintained in a congested network. From the results in this subsection, it can be seen that the proposed method performs better than the other two methods, reduces the average packet delay and jitter, and maintains fairness for UDP uplink flows.

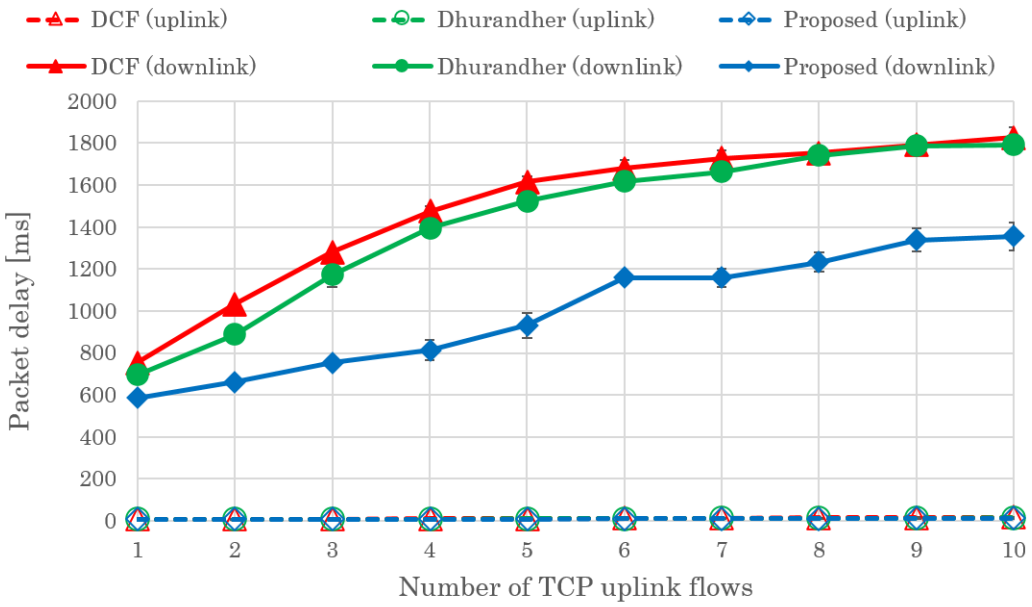

Figure 14: Average UDP packet delay in Scenario 1.

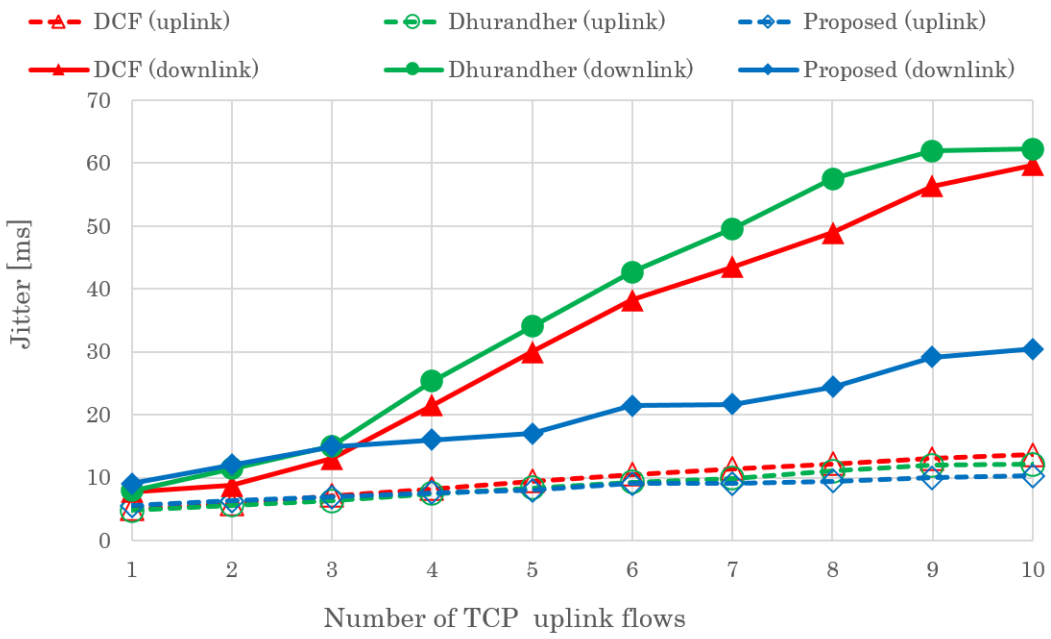

Figure 15: Average UDP packet jitter in Scenario 1.

Table 3: Average fairness index for UDP flows in Scenario 1.

Number of TCP uplink flows	uplink			downlink		
	DCF	Dhurandher	Proposed	DCF	Dhurandher	Proposed
1	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
2	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
3	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.91 ± 0.00	0.96 ± 0.01	1.00 ± 0.00
4	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.76 ± 0.02	0.83 ± 0.01	1.00 ± 0.00
5	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.64 ± 0.02	0.72 ± 0.02	1.00 ± 0.00
6	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.60 ± 0.02	0.66 ± 0.02	0.99 ± 0.00
7	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.57 ± 0.02	0.62 ± 0.03	0.99 ± 0.00
8	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.56 ± 0.02	0.57 ± 0.01	0.98 ± 0.01
9	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.54 ± 0.01	0.56 ± 0.01	0.96 ± 0.01
10	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.54 ± 0.01	0.55 ± 0.01	0.95 ± 0.02

4.3 Simulation results of Scenario 2

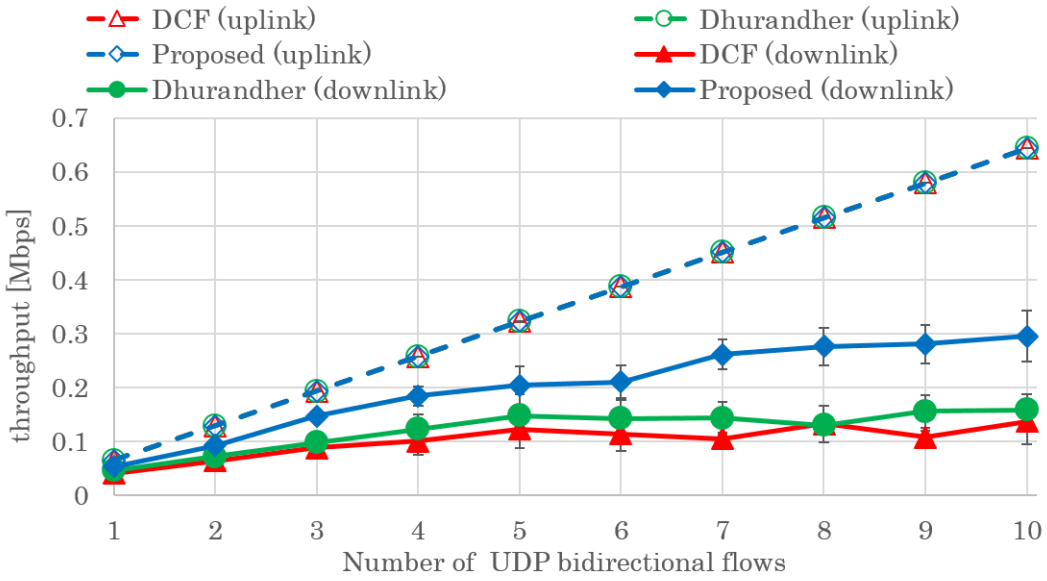

Figure 16: Average total UDP throughputs in Scenario 2.

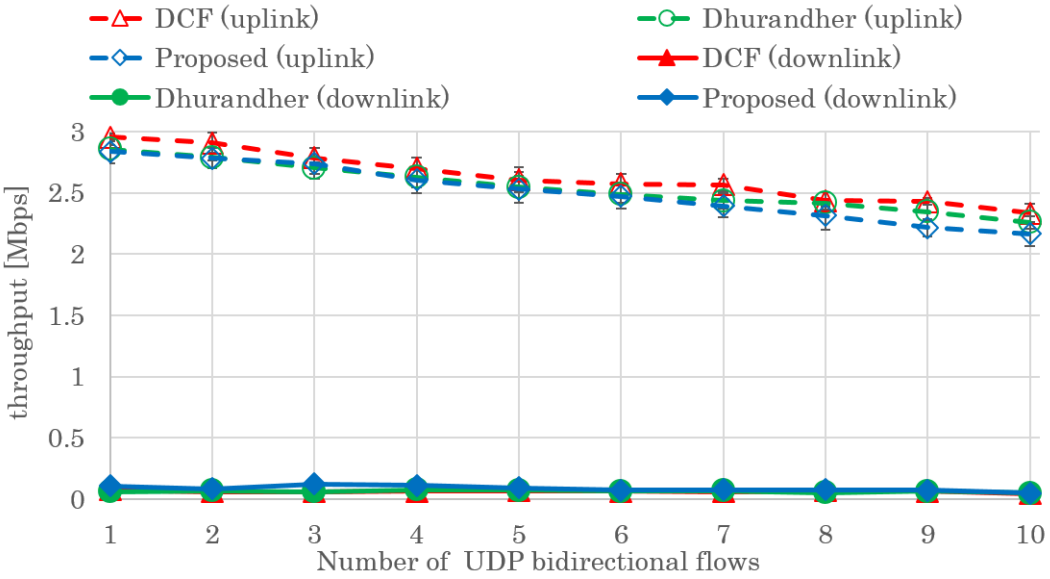

Figure 17: Average total TCP throughputs in Scenario 2.

Figure 16 and Figure 17 show the average total UDP and TCP throughputs. In Figure 16, the throughput of downlink flows in the proposed method is from 0.007 to 0.173 Mbps higher than for the other methods. In Figure 17, the throughput of downlink flows in the proposed method is from 0.001 to 0.0622 Mbps higher.

Figure 18 and Figure 19 show the UDP and TCP packet drop rates. In Figure 18, the throughput of the downlink flows in the proposed method is from 18.29 to 34.93% lower than the DCF and from 9.33 to 28.48% lower than Dhurandher's method. In Figure 19, the throughput of the downlink flows in the proposed method is from 5.25 to 17.39% lower than the DCF and from 1.52 to 13.51% lower

than Dhurandher's method. From the results, although the TCP congestion control may almost stop the communications under the higher packet loss ratio, the collision history has little impact on the proposed method in Scenario 2.

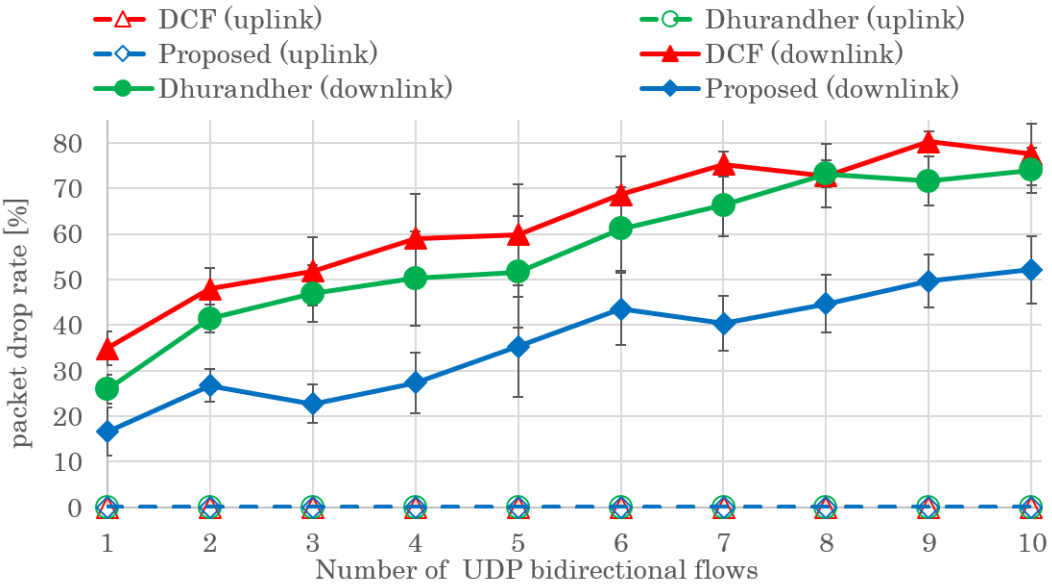

Figure 18: Average UDP packet drop rates in Scenario 2.

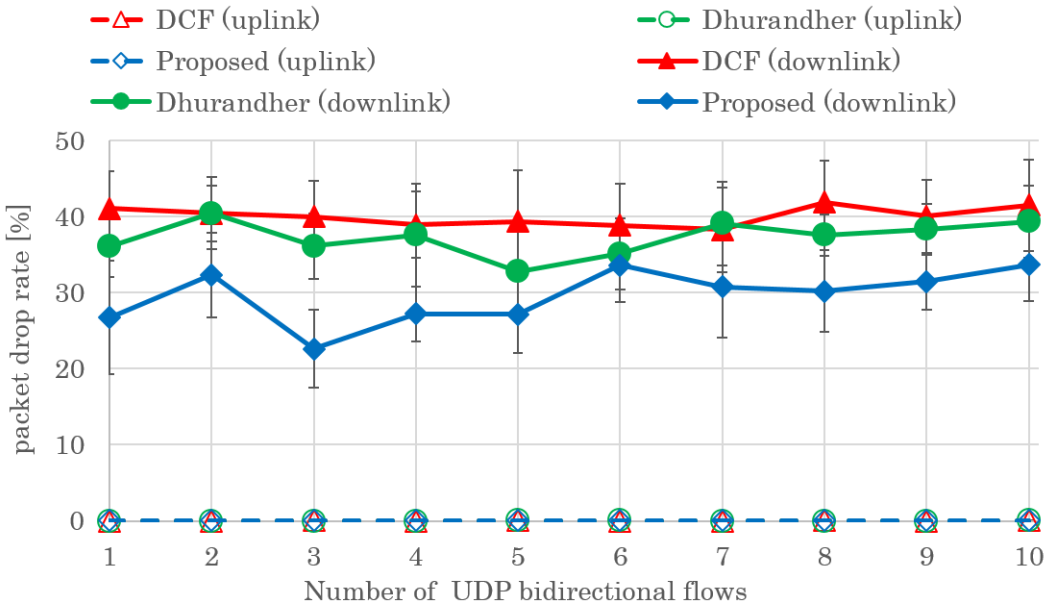

Figure 19: Average TCP packet drop rates in Scenario 2.

To evaluate the QoS of real-time traffic, Figure 20 and Figure 21 respectively show the average UDP packet delay and jitter that allows the QoS in real-time traffic to be evaluated.

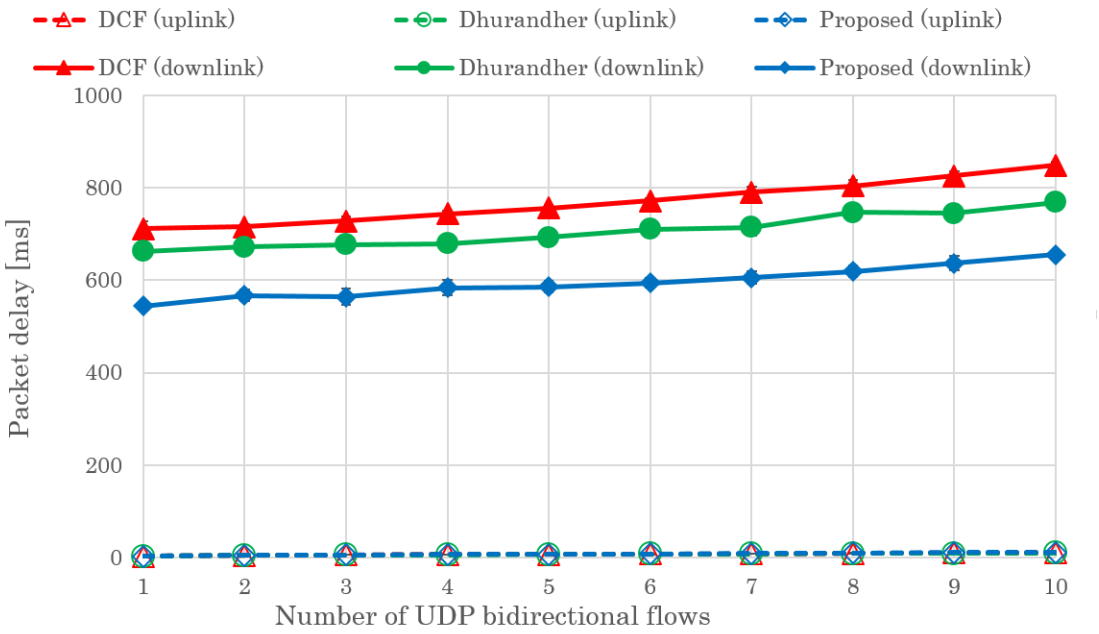

Figure 20: Average UDP packet delay in Scenario 2.

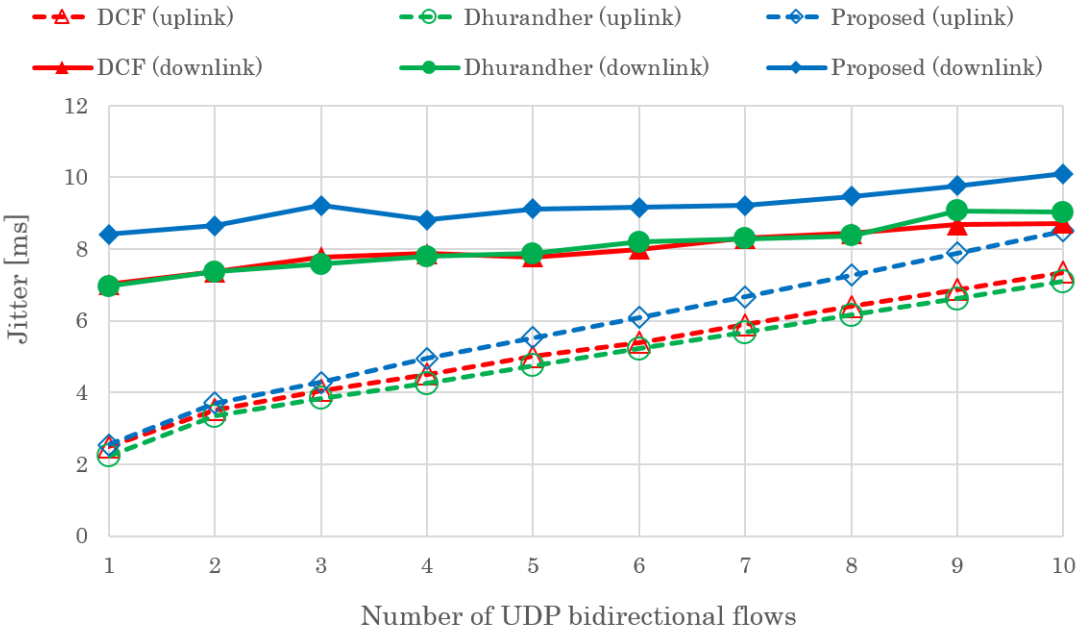

Figure 21: Average UDP packet jitter in Scenario 2.

In Figure 20, the average delay for uplink flows in all the methods was almost 0.0 ms. Although the average delay for downlink flows increased greatly when the number of TCP uplink flows increased, the proposed method improved from 148.3 to 193.9 ms over the DCF, and from 95.4 to 127.3 ms over Dhurandher's method.

In Figure 21, the average jitter of all methods was lower than 10.1 ms. Since the packet interval

of the UDP flow was 40.0 ms, the average jitter was not so large.

Although the average fairness index for UDP flows was also evaluated, that of all the methods was always 1.00 so that the table was omitted. Thus, fairness for each UDP flow was maintained for all methods under this condition. From the results in this subsection, it can be seen that the proposed method gives better average total throughput, packet drop rate, and packet delay for UDP downlink flows than the other two methods.

4.4 Simulation results of Scenario 3

Figure 22 and Figure 23 show the average total TCP throughputs and average packet drop rate in Scenario 3 without UDP flows. The 95% confidence intervals are also provided in both figures, but are too narrow in the former figure to be visible.

In Figure 22, when the number of flows is greater than 1, the average total throughput for all methods is from 10.3 to 12.7 Mbps. However, when the number of flows is 1, those of the DCF and Dhurandher's method are 12.3 Mbps and 11.9 Mbps respectively, but that of the proposed method is 10.3 Mbps. Since the CW range is a little wider for a less congested situation, the total throughput also decreases in the proposed method.

In Figure 23, the average packet drop rate for all methods is approximately 0%, which shows there is less network congestion in Scenario 3. In most cases in Figure 22, the proposed method has less influence on the network condition. However, when only one flow exists, the average total throughput in the proposed method is 1.93 Mbps lower than the other two methods, since, under this condition, the CW range for the lower priority flows in the proposed method is wider than those of the other two methods.

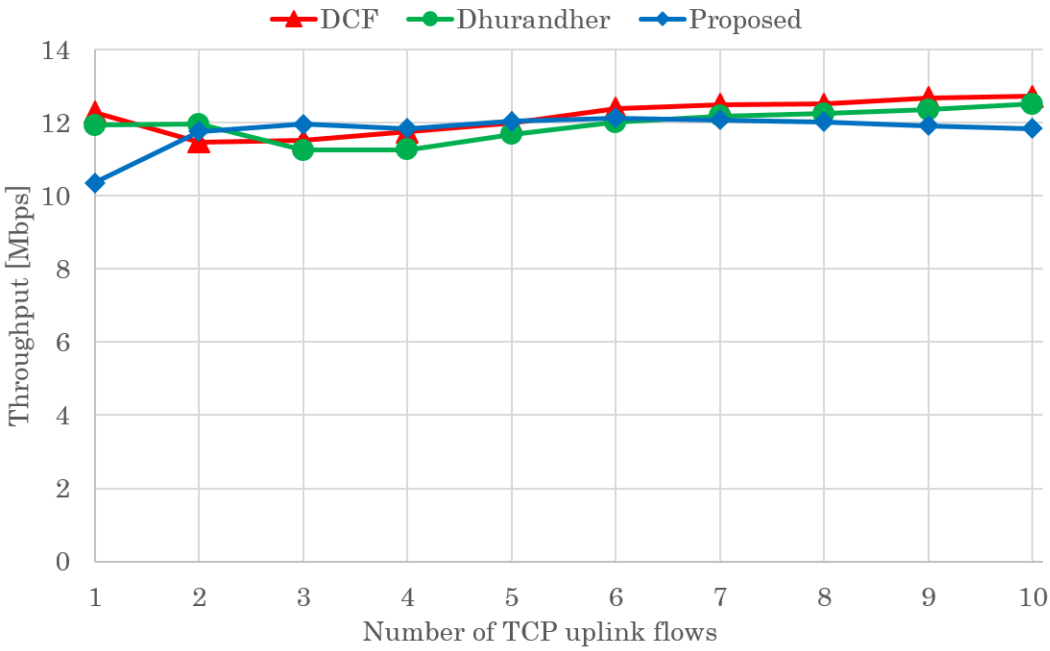

Figure 22: Average Total TCP throughputs in Scenario 3 without UDP flows.

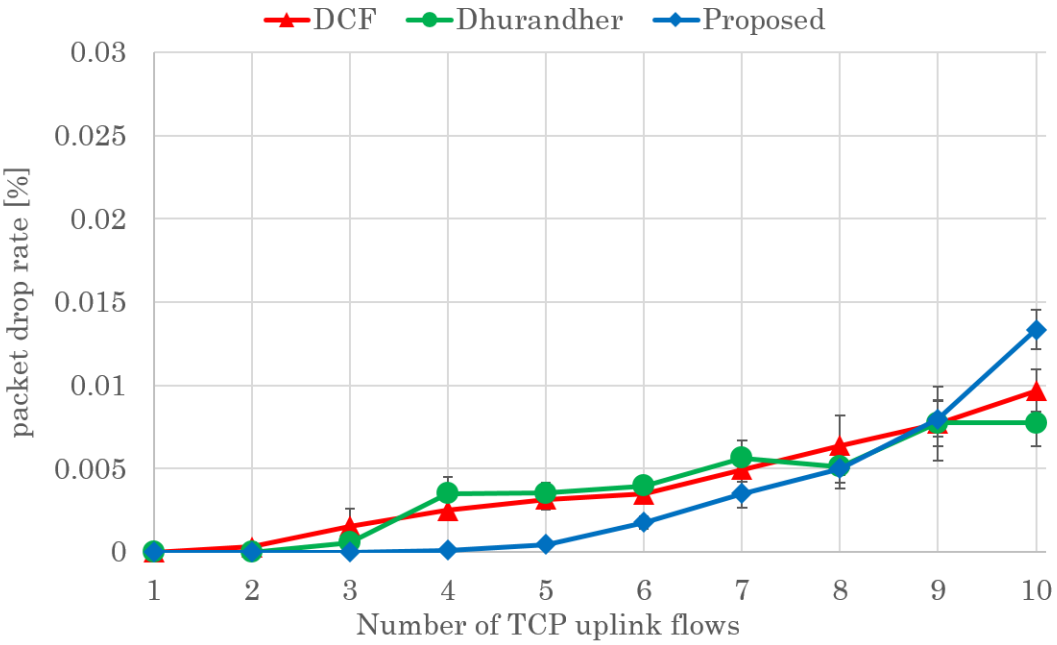

Figure 23: Average TCP packet drop rates in Scenario 3 without UDP flows.

In the next simulation, five bidirectional UDP flows were added to evaluate the performance of high priority flows under light congestion. Figure 24 and Figure 25 show the average total TCP throughputs and average packet drop rate. Figure 26 and Figure 27 also show the average total UDP throughputs and average packet drop rate. Figure 28 and Figure 29 show the average UDP packet delay and jitter. Table 4 shows the average fairness index for UDP flows.

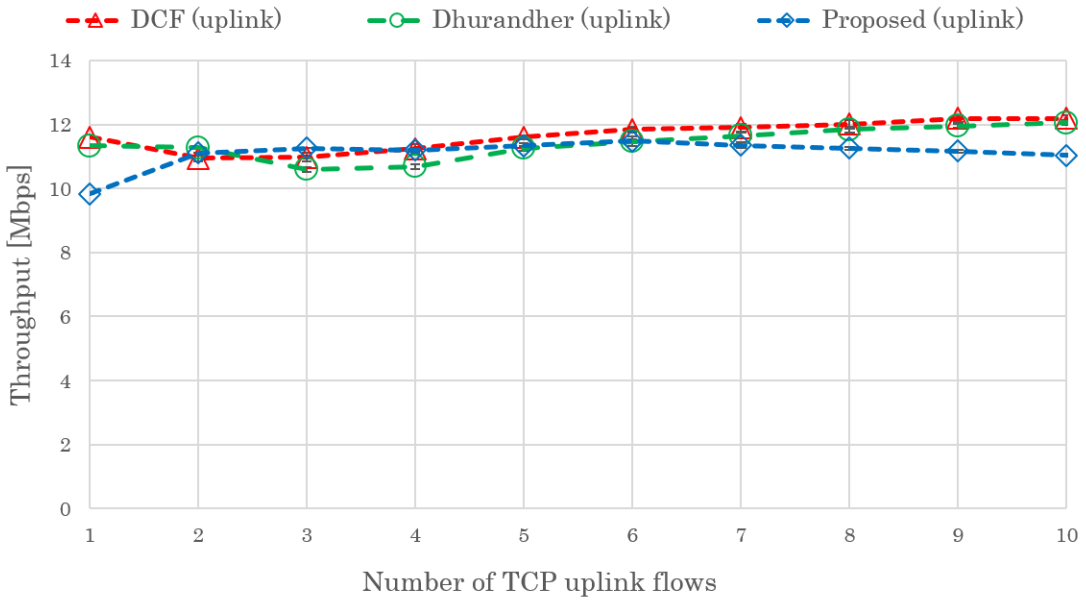

Figure 24: Average Total TCP throughputs in Scenario 3 with five bidirectional UDP flows.

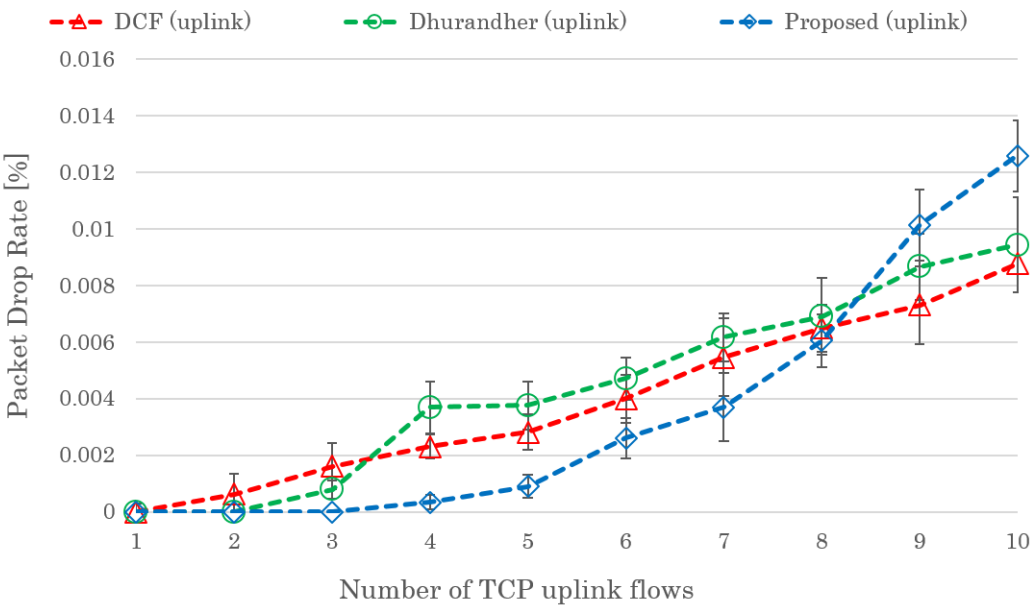

Figure 25: Average TCP packet drop rates in Scenario 3 with five bidirectional UDP flows.

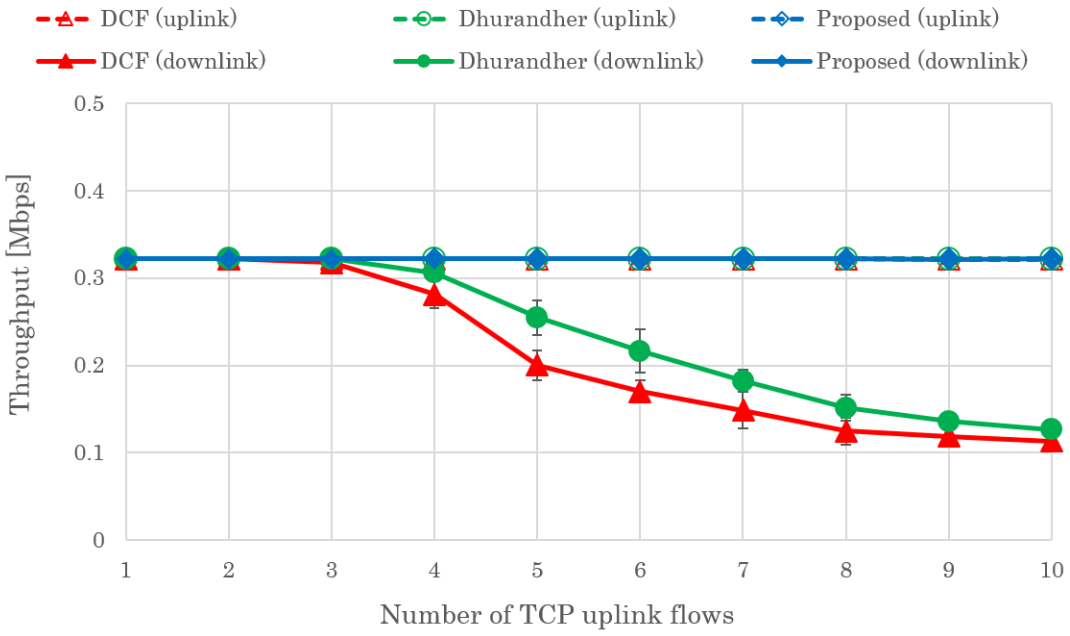

Figure 26: Average Total UDP throughputs in Scenario 3 with five bidirectional UDP flows.

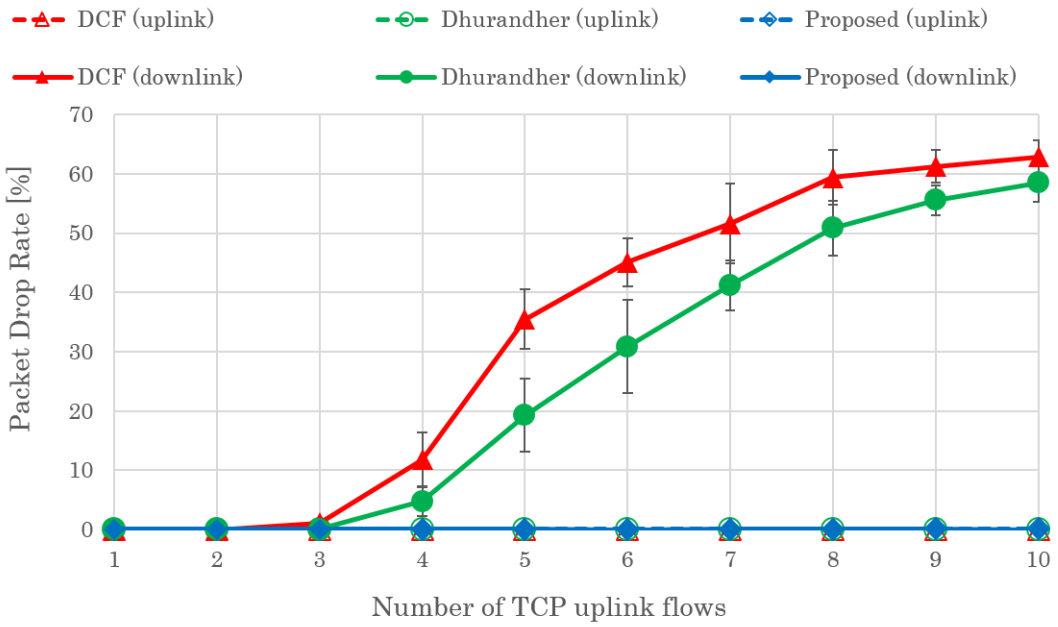

Figure 27: Average UDP packet drop rates in Scenario 3 with five bidirectional UDP flows.

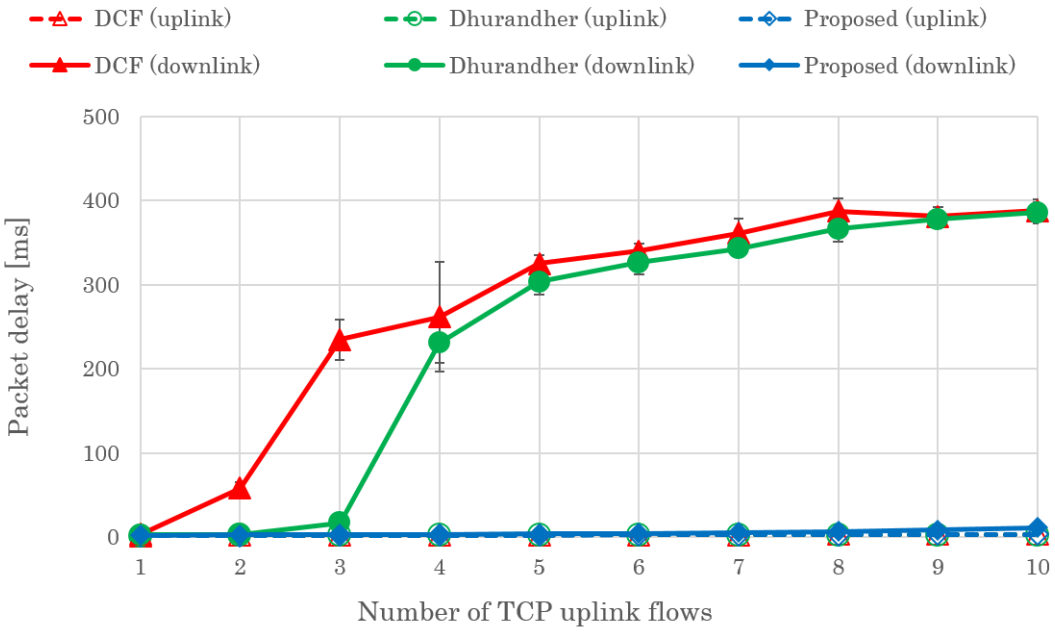

Figure 28: Average UDP packet delay in Scenario 3 with five bidirectional UDP flows.

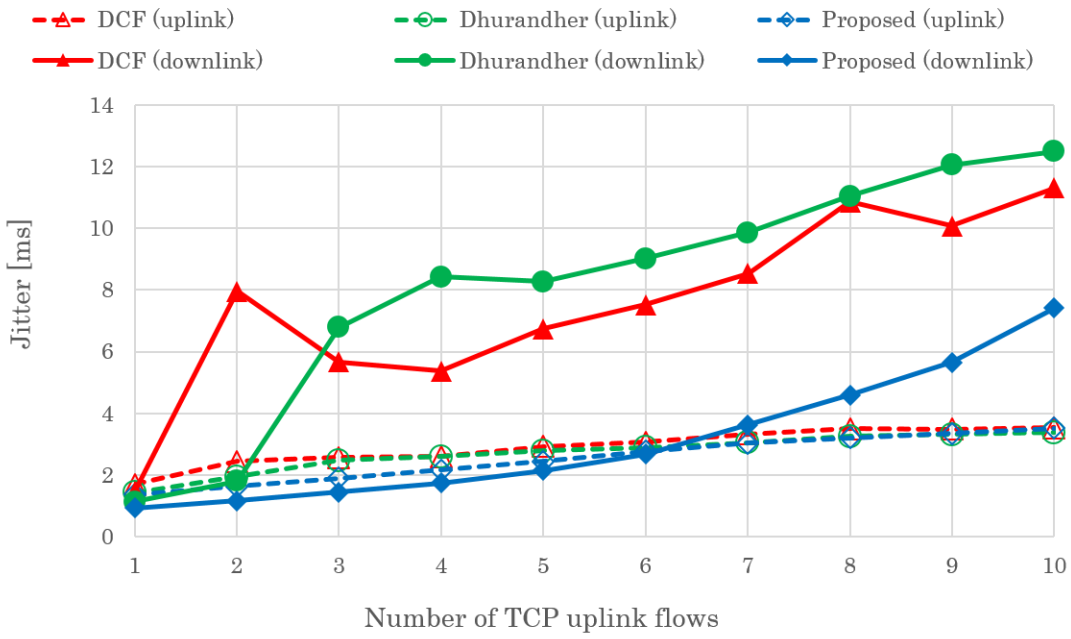

Figure 29: Average UDP packet jitter in Scenario 3 with five bidirectional UDP flows.

Table 4: Average fairness index for UDP flows in Scenario 3 with five bidirectional UDP flows.

Number of UDP bidirectional flows	uplink			downlink		
	DCF	Dhurandher	Proposed	DCF	Dhurandher	Proposed
1	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.71	1.00 ± 0.00	1.00 ± 0.00
2	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.71	1.00 ± 0.00	1.00 ± 0.00
3	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.71	1.00 ± 0.00	1.00 ± 0.00
4	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.70	1.00 ± 0.00	1.00 ± 0.00
5	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.94 ± 0.68	0.98 ± 0.00	1.00 ± 0.00
6	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.90 ± 0.67	0.96 ± 0.00	1.00 ± 0.00
7	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.86 ± 0.65	0.92 ± 0.00	1.00 ± 0.00
8	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.81 ± 0.63	0.87 ± 0.00	1.00 ± 0.00
9	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.77 ± 0.62	0.83 ± 0.00	1.00 ± 0.00
10	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.76 ± 0.62	0.81 ± 0.00	1.00 ± 0.00

In Figure 24, the average total throughput is slightly reduced from that in Figure 22 since the reduced throughput is diverted to the UDP flows. In Figure 25, the average packet drop rate for all methods is approximately 0% as shown in Figure 23.

From Figure 26 and Figure 27, the average total UDP throughputs of downlink flows was maintained at about 0.32 Mbps and the average packet UDP drop rate of downlink flows was almost 0.0% in the proposed method, while the throughputs decreased and the drop rate became higher in the other two methods when the number of TCP uplink flows was greater than 3. In Figure 28, the average UDP packet delay of uplink flows was almost 0.0 ms in the proposed method, while it was more than 200 ms in the other methods when the number of TCP uplink flows was greater than 3. In Figure 29, although the average jitter for all methods was not substantial, that of downlink flows in the proposed method was reduced by more than 3.0 ms compared to that in the other methods. In Table 4, the average fairness index for downlink flows was always 1.00, but that of the other methods became lower than 0.90 when the number of TCP uplink flows was more than 7.

In Scenario 3 with five bidirectional UDP flows, the AP only sends UDP downlink packets and

TCP ACK packets for TCP uplink flows. If UDP packets can always be transmitted before TCP ACK packets arrive at the AP queue, then the delay of UDP downlink packets becomes almost 0.0 ms. In the proposed method, since the CW for TCP ACK packets remained large, UDP packets were always transmitted before TCP ACK packets. However, in the other two methods, some UDP packets could not be sent before TCP ACK packets, then the subsequent UDP packets also stayed with TCP ACK packets in the AP queue. From the results in this subsection, it can be seen that the proposed method does not affect the quality of the high priority traffic in the topology compared to the other two methods.

5 Conclusion

This paper proposed a minimum contention window control method for low priority data based on the collision history for two priority levels and defined the CW range for each priority. Then, the network simulations demonstrated that the proposed method increased the average total throughput of low priority frames and decreased the packet drop rate in both directions compared to the DCF and Dhurandher's method.

For UDP flows, the average delay, jitter, and fairness index also gave better results and showed that the proposed method does not affect the quality of the traffic in the lightly congested network of Scenario 3 to the same extent as the other two methods since the CW for the lower priority downlink flows remains large and that for the high priority downlink flows stays lower so that UDP packets are always transmitted before TCP packets are stored in the AP queue.

However, the proposed method cannot maintain the average total throughput of the downlink traffic even for the high priority traffic in the heavily congested network of Scenarios 1 and 2. Since the AP uses the single tail-drop queue, the AP's transmission opportunity is decreased when the nodes increase their transmission frames. Thus, the proposed method should be introduced as a class-based queue [13] for the AP to achieve fairness for both uplink and downlink in a heavily congested network. In addition, the average total throughput of the lower priority flows also decreased when no congestion occurred. This result also suggests that the CW range of these flows in the proposed method should be reduced to improve their average total throughput.

In the simulations, all nodes were assumed to support the same frame transmission control protocol such as Dhurandher's method and the proposed method. However, it should be verified in the case where the proposed method and other access control methods are mixed. For example, if many nodes attempt to send frames and congestion occurs, then the throughput of low priority in the proposed method may be smaller than that of the DCF when these two methods are mixed because the DCF's CW range at the initial transmission is smaller than that of the low priority in the proposed method. In the case that IEEE 802.11 Enhanced Distributed Channel Access (EDCA) is also adopted, the throughput of its high priority traffic might be significantly higher than that of the proposed method since its IFS is smaller than DIFS. On the other hand, when heavy congestion occurs, the EDCA's throughput may be lower than that of the proposed method. For the high priority, the CW range of the proposed method is always half of the current CW range or stays at CW_{min} if the current one is already at CW_{min} after a successful transmission.

In the simulation conditions, voice flows and data flows are the assumed communication types. However, other types of communication that require more flexible priority control should be considered. For this purpose, the proposed method should be expanded to support more than two priority levels.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP26280027.

References

[1] Teerapat Sanguankotchakorn, Aravinthan Gopalasingham, and Nobuhiko Sugino. Adaptive channel access mechanism for real time traffic over IEEE 802.11e Wi-Fi network. In *Proceedings of International Conference on Intelligent Systems, Modelling and Simulation*, pages 486–491, 2013.

[2] Katarzyna Kosek-Szott, Marek Natkaniec, and Lukasz Prasnal. IEEE 802.11aa intra-AC prioritization — a new method of increasing the granularity of traffic prioritization in WLANs. In *Proceedings of IEEE Symposium on Computers and Communication (ISCC)*, pages 1–6, 2014.

[3] Mehaseb Ahmed, Gamal Abdel Fadeel, and Ibrahim Ismail Ibrahim. Differentiation between different traffic categories using multi-level of priority in DCF-WLAN. In *Proceedings of International Conference on Telecommunications*, pages 263–268, 2010.

[4] Nao Emoto, Shigetomo Kimura, and Yoshihiko Ebihara. Evaluation of random discard method on wireless LAN APs for real-time communications (in Japanese). In *Proceedings of the 74th National Convention of IPSJ*, pages 507–508, 2012.

[5] Sanjay K. Dhurandher, Issac Woungang, Sahil Sharma, and Veeresh Goswami. A priority based differentiation for contention mechanism in legacy DCF method. In *Proceedings of International Conference on Advanced Information Networking and Applications Workshops*, pages 478–482, 2013.

[6] Abhay K. Parekh and Robert G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM transactions on networking*, 1(3):344–357, 1993.

[7] Jon C. R. Bennett and Hui Zhang. WF^2Q: worst-case fair weighted fair queueing. In *Proceedings of IEEE INFOCOM'96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation*, volume 1, pages 120–128, 1996.

[8] C. Mala and B. Nithya. Dynamic sliding contention window adjustment in saturated wireless networks. In *Proceedings of International Conference on Network-Based Information Systems*, pages 186–193, 2014.

[9] Ikram Syed, Bosung Kim, Byeong hee Roh, and Il hyuk Oh. A novel contention window backoff algorithm for IEEE 802.11 wireless networks. In *Proceedings of IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS)*, pages 71–75, 2015.

[10] Nao Emoto, Shigetomo Kimura, and Yoshihiko Ebihara. IEEE 802.11 contention window control method based on the number of wireless terminals to improve transmission probability for APs (in Japanese). In *Proceedings of the 72th National Convention of IPSJ*, pages 167–168, 2010.

[11] Hongxian Tian and Wei Yang. Study on QoS of multimedia traffics in MAC layer based on 802.11. In *Proceedings of International Conference on Information Science and Applications (ICISA)*, pages 1–6, 2012.

[12] Pedro Fortuna Gustavo Carneiro and Manuel Ricardo. Flowmonitor: a network monitoring framework for the network simulator 3 (ns-3). In *Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools*, volume 1, pages 1–10, 2009.

[13] Sally Floyd and Van Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM transactions on Networking*, 3(4):365–386, 1995.