CHAPTER 2

Related Work

There are several performance studies about Markov Chain in 802.11 [6][7][8] or 802.11e [9][10][11]. [7][8][9][10] are all inspired from the same Markov chain in [6]. These studies assumed each station in DCF or QSTA in EDCA has only one flow, and every flow has fully loading and always has a packet. In such saturated network environment, we can not observe QoS-sensitive traffics without greediness precisely, and it was not considered that one QSTA usually has more than one traffic flow either. The [11] considers the channel status and virtual collision between four Access Categories.

There are several Token Bucket mechanism and Call Admission Control mechanism in 802.16 PMP [12] 802.16 mesh mode [13]. [11][12][13] are simulated 802.11e and 802.16 actions, and obtained the system result such as delay and throughput. The general results come from experiments in network simulators.

2.1. Markov Chain Studies about 802.11e

The authors of [6] provided a bi-dimension discrete-time Markov chain to simulate the backoff action in the 802.11 DCF. Each bi-dimension process represses one station. The first process represses the backoff stage, which influences the maximum value of second process- backoff number. In backoff time interval, the second process will counts down continued for each idle slot. When the second process becomes zero, this station will transmit packet, and it will cause a collision or a success transmission. After a transmission or a collision, the bi-dimension process will draw a random number which is between 0 and W_i-1. W is an abbreviation of CW, and i means the first process value. Since the Markov chain and transition probabilities are regular, the authors solved it in saturation conditions. The authors assume that all transmission probabilities of any station at any idle slot are the same. Here, the idle slot means the medium is sensed idle for a DIFS. With the fully-load traffic, the transmission probabilities of any station at the same idle slot should be equal, and the transmission probabilities of any station at different idle slots vary. But in [6], the authors assume that the transmission probabilities of any station at different idle slots are the same, and it may influence the degree of accuracy.

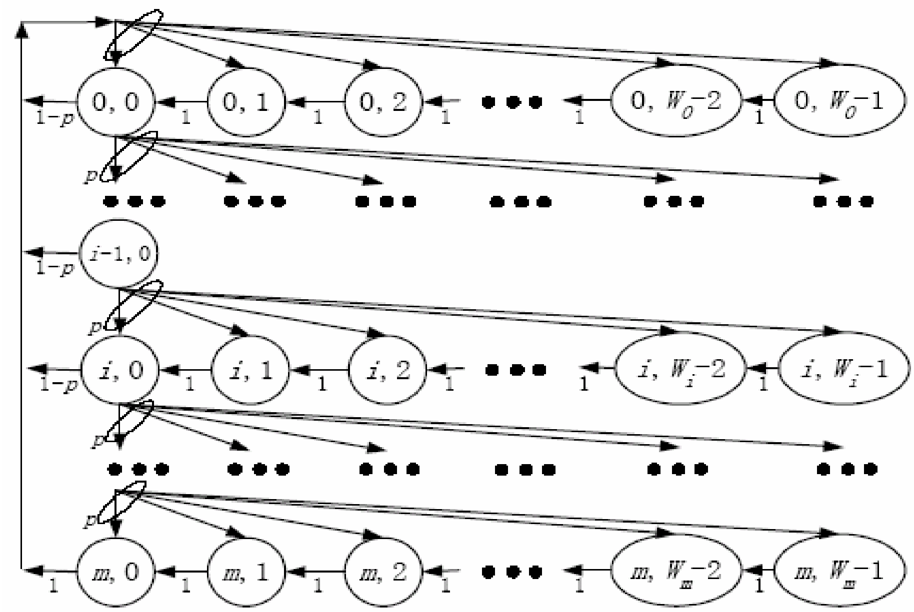

Figure 2-1: The Modified DCF Markov Chain in Backoff Period.

The authors of [8] also followed Markov chain in [6] and compared different CW for different service.

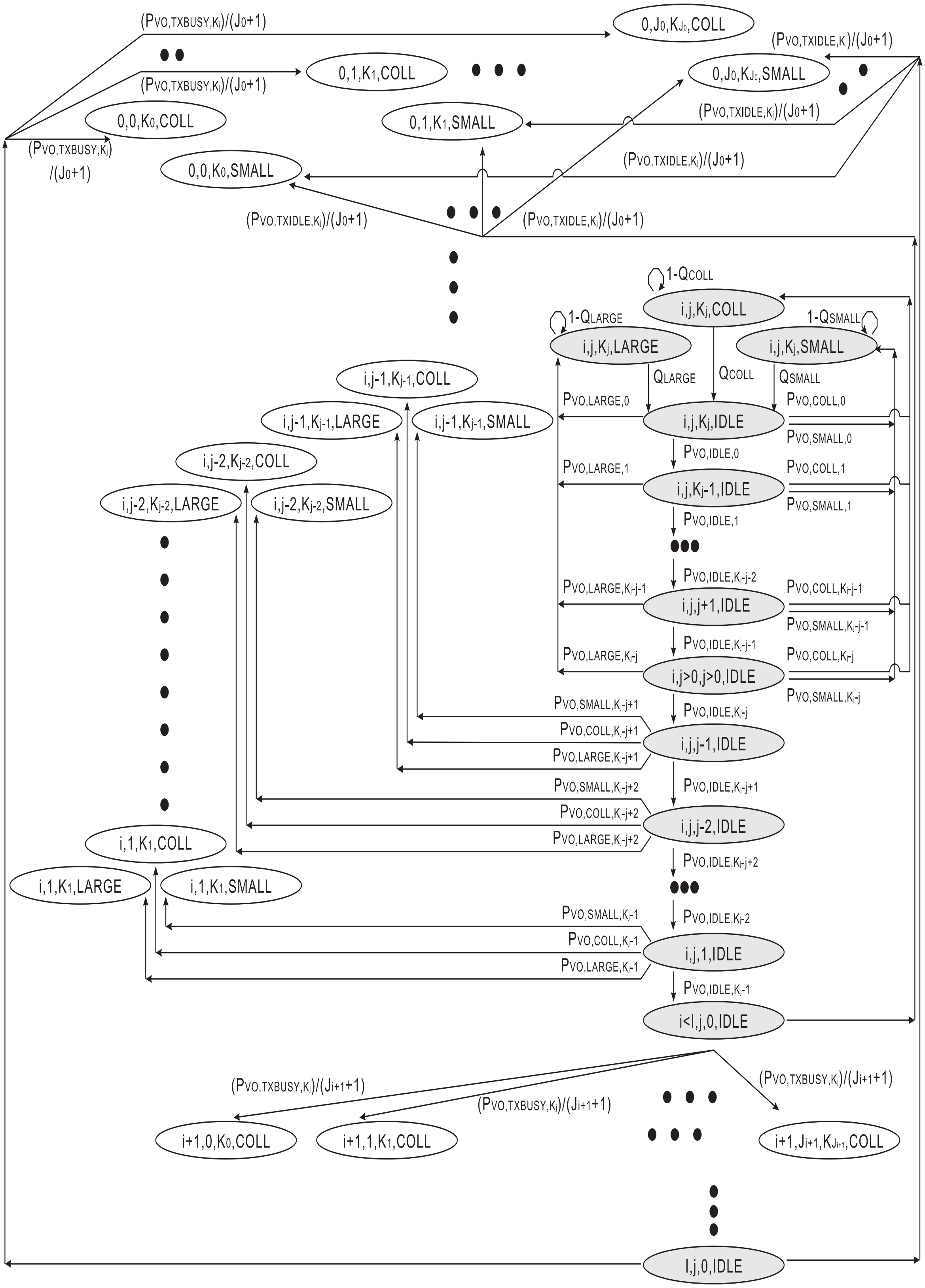

Figure 2-2: The EDCA Markov Chain for VO.

In [11], there are four-dimensions in the Markov Chain model. Retry limit of 802.11 is I. Let i(t) be the first discrete-time stochastic process representing backoff stage. The second discrete-time stochastic process is j(t). It represents the backoff number not including CWO at the last time which the channel is sensed busy. The upper bound of j(t) defined is J_i which depend on the value of i(t). $J_{AC,i}$ (J_i) is the CW_{AC} of backoff stage i. The CWO_{AC} means the result of subtracting self AC's AIFSN off the minimum AIFSN of all ACs. Let k(t) be the backoff number including CWO_{AC}. When it counts down to zero, it will try to access the medium. The last stochastic process s(t) represents the channel state. There are four channel states. IDLE means the channel is idle at least for a PIFS time. SMALL, LARGE, COLL mean the channel is sensed busy because of VO transmission, other AC transmission, or collision respectively.

2.2. Token Bucket and Call Admission Control about 802.16

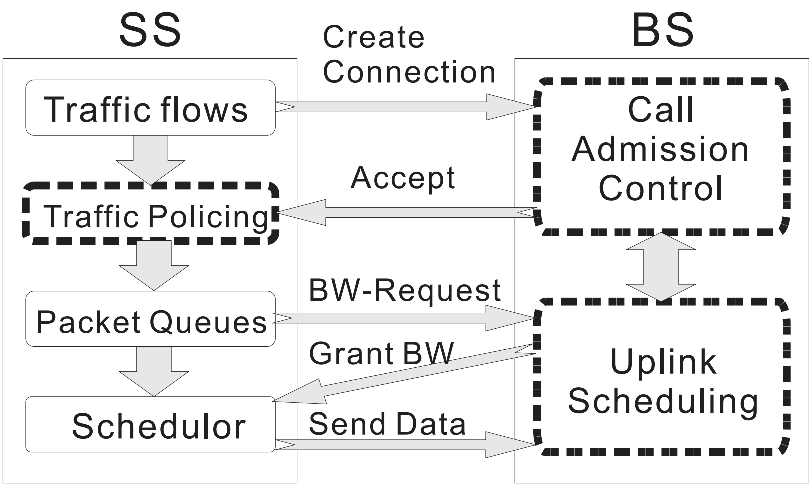

Figure 2-3: IEEE 802.16 PMP operation process.

In [12], a mathematical mode in characterizing traffic flows is also proposed. In figure 2.3, the operation process is depicted, and the cubes with dotted-lines are not defined in the IEEE 802.16 standards.

In this paper, the bandwidth is estimated by the token bucket mechanism, and the Call Admission Control is implemented using these estimated bandwidth information. The principle of the CAC algorithm is: First, the system calculates the current available bandwidth. Second, for the new incoming flows, the system estimates the bandwidth it will take and the system will decide to grant this new flow or not.

Considering an rtPS connection i with parameters r_i, b_i and d_i, which represent its token rate (1/sec; generating a packet consumes a token), bucket size (number of tokens), and delay requirement respectively. During a time period t, a connection controlled by token bucket mechanism with token rate r_i and bucket size b_i can generate $r_i t+b_i$ packets at most. Let f be the duration of an 802.16 super frame. Let the delay requirement of the rtPS connection, d_i, be at least three super frame durations and less than four super frame durations. That is, $4f > d_i \geqq 3f$. Assume that the rtPS connection has a transmission burst, which begins at time t and finishes at time t+6f. The time period of this burst is 6f, so we can know the maximum size generated by this rtPS connection during this burst is $6r_i f+b_i$.

The scheduling is done on a per super frame basis, therefore the precision of the scheduling is a super frame. For this reason, we take 3f as the delay requirement of the rtPS connection, e.g. $\left\lfloor \frac{d_i}{f} \right\rfloor = 3$. If there are n packets of the rtPS connection arrival at time t, and the delay requirement of the rtPS connection is d, the deadline of these packets is t+d-f. Apply the result to the present case, we can find that if there are n packets of the rtPS connection arrival at time t, the deadline of these packets is $t + \left\lfloor \frac{d_i}{f} \right\rfloor - f$, which equals to t+2f. So the deadlines of the packets arrival at time t, t+f, …, t+6f are t+2f, t+3f, …, t+8f, respectively. This means that the packets arrival at the time interval [t, t+f] should be transmitted at the time interval [t+2f, t+3f] at latest. Now we assume tokens stocked in the bucket are ran out at time interval [t+2f, t+3f], Figure 3.1 shows the amount of packets that should be transmitted according to their deadlines during this burst of the rtPS connection.

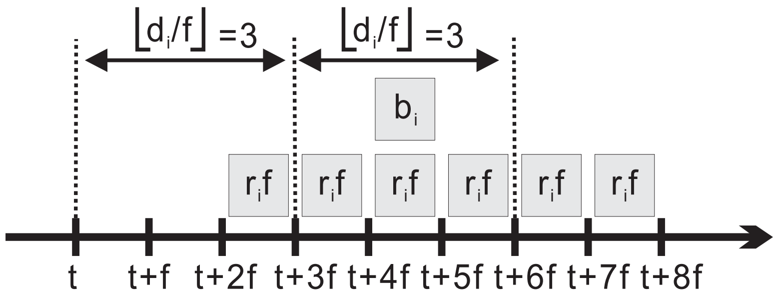

Figure 2-4: The amount of packets that should be transmitted according to their deadlines.

From figure 2.4 we know that we should give the rtPS connection the capacity of $r_i f+b_i$ packets at the time interval [t+4f, t+5f]. But we don't know that when the tokens stocked in the bucket are ran out, we must give the rtPS connection the capacity of $r_i f+b_i$ packets at any time intervals for guaranteeing the delay requirement. These $r_i f+b_i$ packets come from the time interval [t+2f, t+3f], so we also can give the rtPS connection the capacity of $r_i f+\frac{1}{2}b_i$ at the time intervals [t+3f, t+4f] and [t+4f, t+5f] separately, such as Figure 2.5.

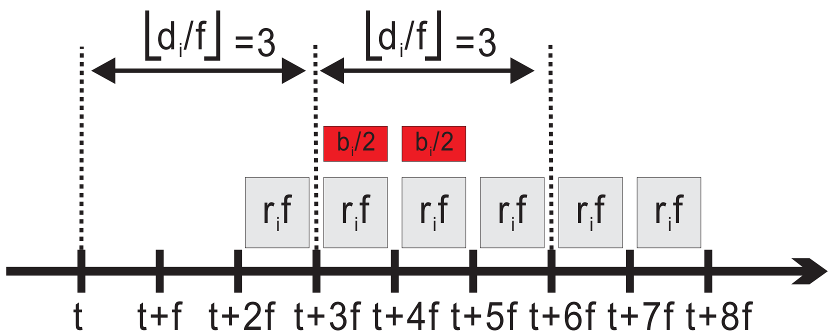

Figure 2-5: Sharing the b_i packets from time interval [t+2f, t+3f] at time intervals [t+3f, t+4f] and [t+4f, t+5f].

If we use the allocation method above, we can only give the rtPS connection the capacity of $r_i f + \frac{1}{2} b_i$ at any time intervals. Actually, if $\left\lfloor \frac{d_i}{f} \right\rfloor$ equals to m_i, we should only give $r_i f + \frac{1}{m_i - 1} b_i$ to the rtPS connection for guaranteeing the delay requirement because there are m_i-1 time intervals that can share the b_i packets. Hence we know the maximum bandwidth requirement of a rtPS connection i, which has parameters: token rate r_i, bucket size b_i, and delay requirement d_i, is

$$r_i f + \frac{b_i}{m_i - 1}, \text{ where } m_i = \left\lfloor \frac{d_i}{f} \right\rfloor \text{ and } d_i \geq 2f$$

The reason for $d_i \geq 2f$ is that the packets arrival at this frame can only be sent after this frame due to the scheduling of BS.