ECEN 689 – Real-Time Wireless Networks: Project 3 Smart Scheduling with Point Coordination Function for WiFi Uplink Transmissions

Ping-Chun Hsieh
lleyfede@tamu.edu

Tao Zhao
alick@tamu.edu

Dongni Han
handongni2015@tamu.edu

Due on 4/6

Terminology

In our report, we use AP or "server" to denote the WiFi access point, and "client" to denote the terminal device such as a mobile phone, a tablet, and so on. Throughout our simulation, we let node 0 be the server, and the other nodes be the clients. The basic time unit for packet transmission is called *slot*, which should be greater than a round-trip time (RTT). For real-time traffic, we group an integer number (denoted by T) of slots into an *interval*, which is the relative deadline of the packets.

1 Background

We consider a wireless network of one AP and N clients, where $N \geq 1$. We consider uplink transmission with point coordination function (PCF). For uplink transmission, packets are generated by each client n in each interval. Number of packets X_n follows uniform distribution $\{ U_{min}, U_{max} \}$. And we only consider real-time traffic. In PCF mode, AP polls at most one client per slot and it has full control over channel allocation. In the beginning of each interval, each client generates a random number of packets and waits for channel access that is fully controlled by the AP. Since the packets are generated and queued on the client's side, the AP needs to collect the queue information of the clients to make scheduling decisions.

With a baseline policy, in the phase 1, the AP collects the queue information by polling each client one by one for the number of data packets it generates (X_n). After all clients are polled, the AP knows the number of packets at each client. In the phase 2, the AP uses Max-Weight scheduling to choose one of the clients in each time slot for actual data transmission. However, there are three issues about baseline policy. Firstly, no data transmissions is in phase 1. Secondly, channel utilization for data packets is very low. Thirdly, overhead is especially low when the number of clients N is large or the channel is very poor because the AP spends much time on polling number in phase 1.

2 Smart Policy

We use three different features to improve performance under symmetric and asymmetric channel. All combinations of three features are possible. And we use binary to represent those combination.

Table 1: Policy Types

Policy Types	Binary representation
Baseline	000
Selective	001
Piggyback	010
Selective + Piggyback	011
Retry Limit	100
Selective+ Retry Limit	101
Piggyback + Retry Limit	110
Smart	111

2.1 Selective Polling

The first feature is to use selective polling. In each interval, the AP only polls $n \leq N$ clients. The process of determining the optimal n is following:

1. Sort clients by channel reliabilities $p_1 > p_2 > \cdots > p_N$
2. Estimated throughput: $\hat{R}_n = \min\{n\overline{U}, (T - \sum_{i=1}^{n} \frac{1}{p_i})\frac{\sum_{i=1}^{n} p_i}{n}\}$
3. Find the optimum $n^* = argmax_n \hat{R}_n$

The system uses random permutation for fairness by using Knuth shuffle algorithm. And only after all selected clients are served completely, we consider remaining clients by serving clients one by one.

2.2 Piggybacked Queue Length

The second feature is to use piggybacked queue length. It means that clients reply both its number of packets X_n and first data message in one packet to the AP. The process of determining the optimal n is following:

1. We need define new packets types: `SWiFi_PKT_POLL_PGBK` and `SWiFi_PKT_PGBK_UL`.
2. If it is combined with selective polling,
   (a) All slots are effectively available for data.
   (b) Its estimated throughput is $\hat{R}_n = min\{n\overline{U}, T\frac{\sum_{i=1}^{n} p_i}{n}\}$.

2.3 Retry Limit for Polling

The third feature is to use retry limit for polling. By using this feature, it can avoid spending too much time on clients when the channel is poor or the number of clients is large. The process of determining the optimal n is following:

1. Set `num_retry_ = 0` for the newly polled client
2. If retry limit reached: poll the next client in the next slot regardless the last transmission is successful or not
3. Otherwise: `num_retry_++`

2.4 Smart Policy

Our smart policy combines selective polling and piggybacked queue length and retry limit for polling together.

1. Select a subset of clients to poll X_n.
2. AP polls clients in expect of piggybacking reply.
3. AP repeats polling a client for limited times.

3 Implementation in NS-2