**EECE.7290 Special Topics on SDN and Data Plane Programming**

**Lab6**

**Introduction:**

In the previous labs, we learned how to correctly setup and run DPDK application. But, we have not combined DPDK with a real problem. In this Lab6, we are going to integrate the streaming algorithm Count-Min sketch with DPDK to solve the heavy-hitter problem.

**Tasks:**

1. Read <http://web.stanford.edu/class/cs369g/files/lectures/lec7.pdf> to understand the algorithm of the Count-Min sketch. The original paper can be found at <http://dimacs.rutgers.edu/~graham/pubs/papers/cm-full.pdf>

2. Based upon the main.c of Lab2-RX, add the code which implements the Count-Min sketch into the main.c such that the new code can detect and print the heavy-hitter. In this Lab6, the flow is defined as the 5-tuple (src IP address, destination IP address, src port, destination port, and protocol), and heavy-hitter is defined as the 5-tuple which has the most number of packets.

3. The **pseudo-code** of Count-Min sketch is shown below, and you need to complete the detailed definition of “struct Flow” , “hash0”, “hash1”, and “hash2”.

struct Flow;

struct Heavy\_Hitter {

struct Flow flow;

uint32\_t count;

};

//Gloabal

struct Heavy\_Hitter heavy\_hitter;

uint32\_t sketch[3][N];

uint32\_t hash0(struct Flow flow);

uint32\_t hash1(struct Flow flow);

uint32\_t hash2(struct Flow flow);

//

foreach flow in flow\_set {

uint32\_t hv[3];

uint32\_t hv[0] = hash0(flow);

uint32\_t hv[1] = hash1(flow);

uint32\_t hv[2] = hash2(flow);

for(i=0; i<3; i++) {

sketch[i][hv[i]] += 1;

}

uint32\_t min = sketch[0][hv[0]];

for(i=1; i<3; i++) {

if (min > sketch[i][hv[i]]) {

min = sketch[i][hv[i]];

}

}

if(min > heavy\_hitter.count) {

heavy\_hitter.flow = flow;

heavy\_hitter.count = min;

}

}

**Submissions:**

A single report which includes the following

1. Explain the Count-Min in your own words and how you implement it in the main.c.

2. Attach your modified main.c and indicate where the Count-Min is implemented.