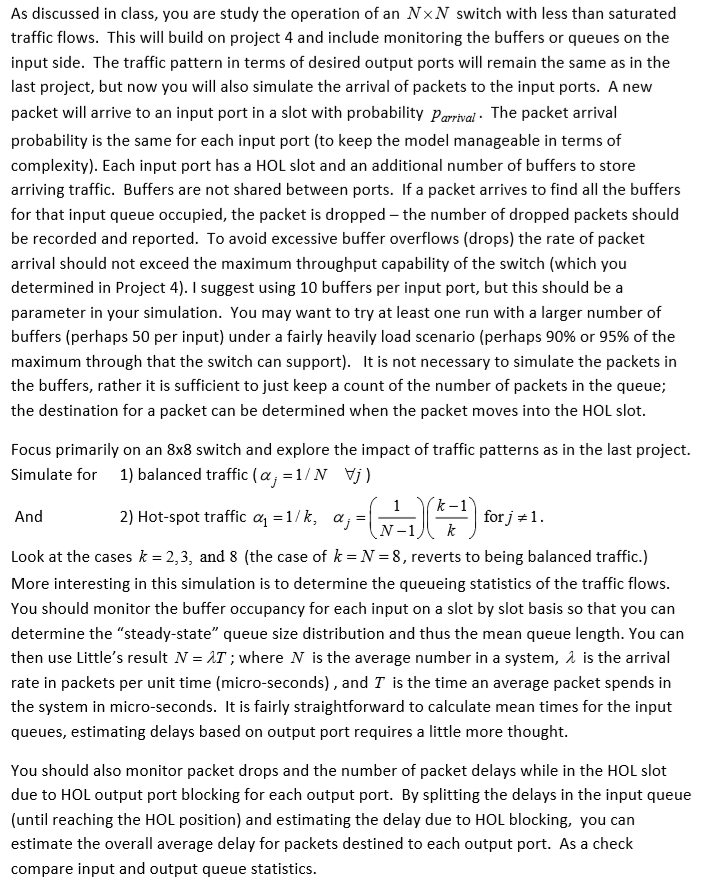
**Project 5：Switch Performance including Buffering**

**EE 511 – Section** Tuesday 5:00pm—5:50pm

**Name: Yiqi Feng**

**Student ID #:** 9057129035

***Problem Statement***



***Theoretical Analysis***

In project 4, we’ve got the throughput and assumed that every input port at one time will have a new arriving packet.

In this project, we suppose the packet arrived with probability Parrival. And then do the following analysis

First of all, the mean queue length for each input port. In the simulation, we will record the queue length for each input port in every time step. And ﬁnally, we can calculate the average value of it to get the mean queue length. We can also count the number of different length of queue to get the queue size distribution.

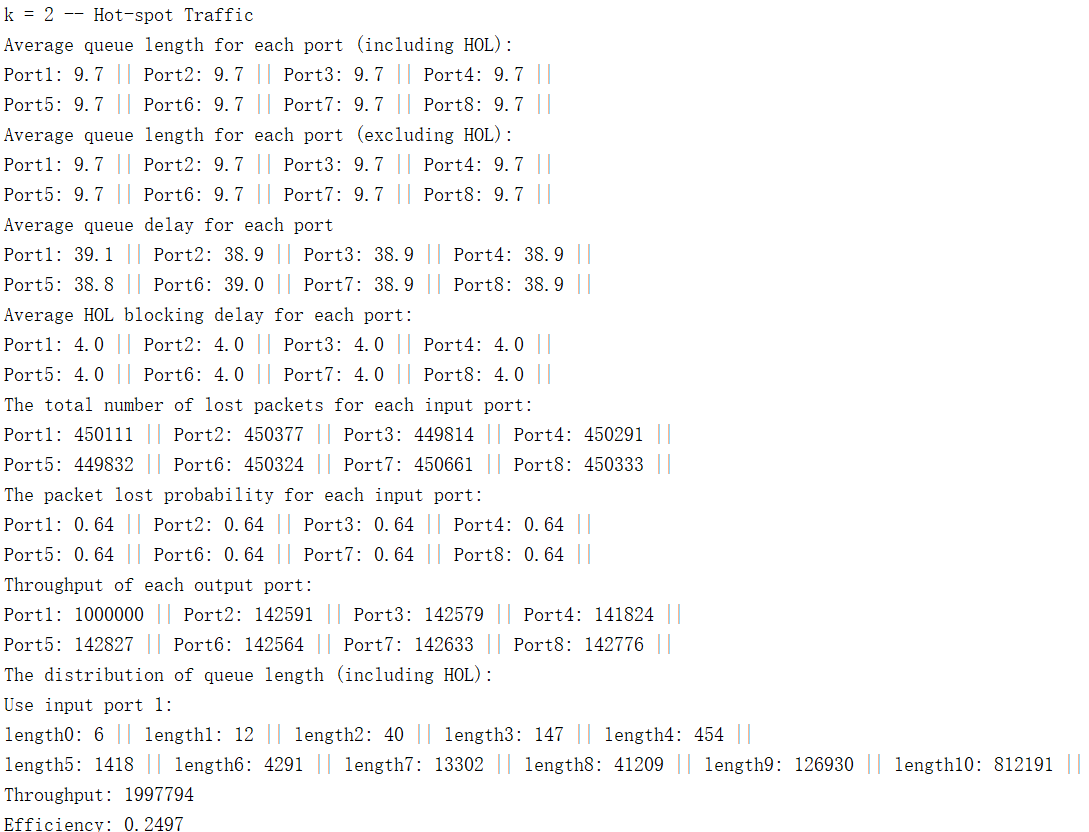
Then, the mean buffer/queue delay for each input port. Before reaching the HOL position, the packet will stay in the queue for a period of time. To estimate this delay, I count the throughput for each input port. Then I divide the simulation time by the throughput of each input port. Thus, I get the time interval between two consecutive packets, and then if I multiply it by the mean queue length, I can get the approximation of mean queue delay. To sum up, We can calculate it in a formula below:

In this function, N is the mean queue length, λ means the arrival rate of packets, and T stands for the average time packets stay in the system, which is the mean queue delay. So if we can get N and λ, we can get the mean queue delay.

Moreover, the HOL blocking delay, which is the second part of the total delay. After reaching the HOL position, because there may be other packets which are destined to the same output port, the packet sometimes need wait. We can use time interval between two consecutive packets to estimate it, which is:

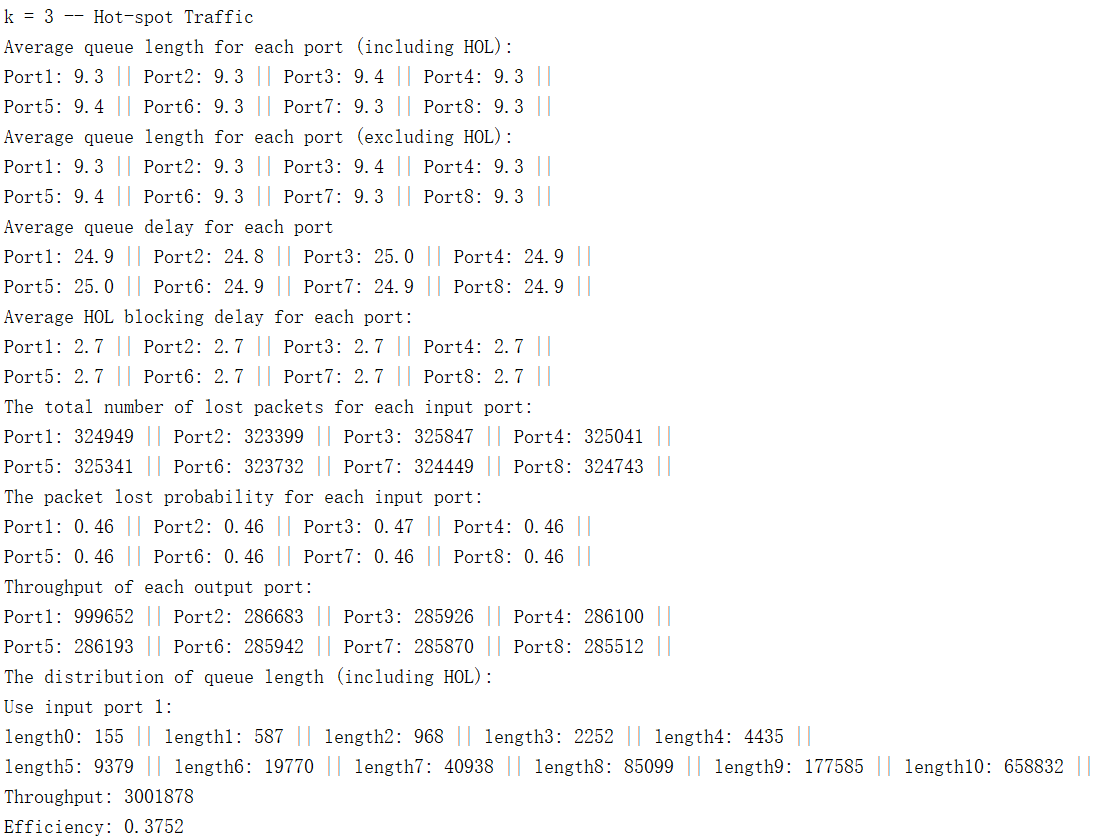
At last, the number of lost packets of each input port. Because of limited number of the buffer size, some packets are dropped by the switch. It is straightforward to estimate this statistic, every time once a packet get dropped, we use a counter to record it. Then, we divide the total number of lost packets by the total number of arrived packets, we can get the packet loss rate.

***Experiments and Results***



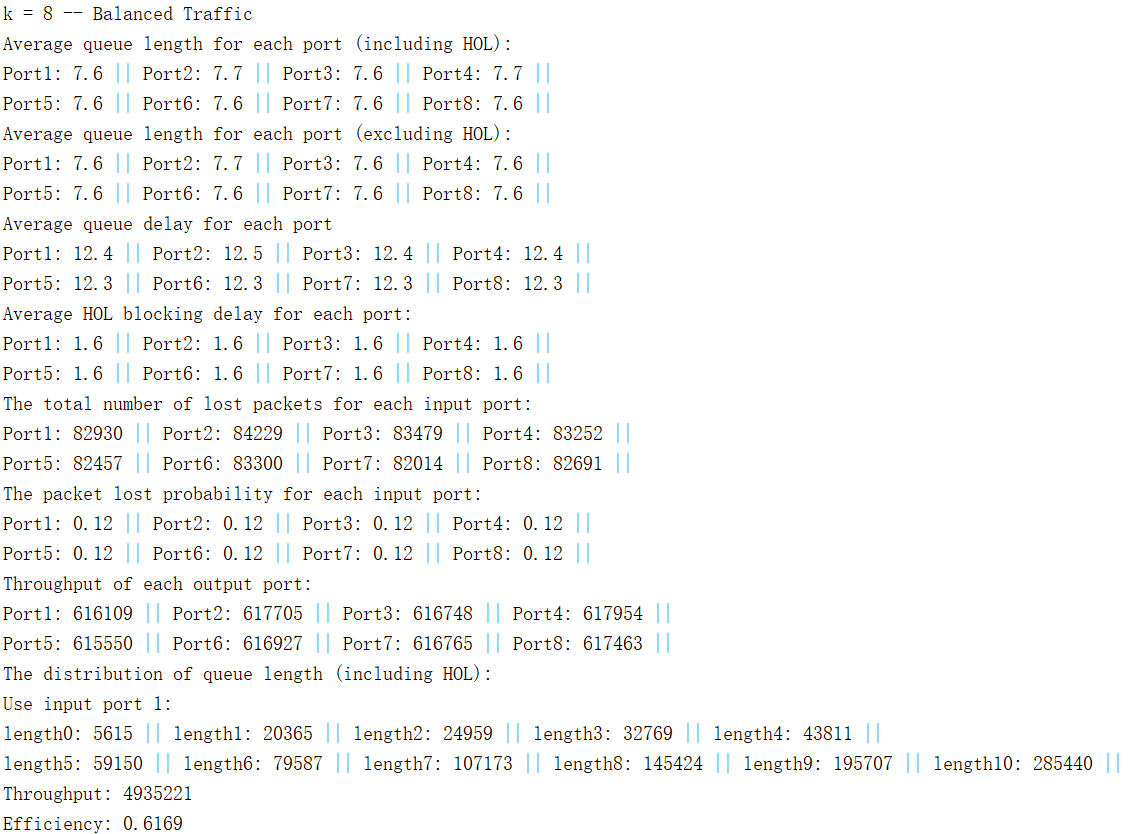
In this case, T is about 38, and the λ = Parrival ×(1−Plost) = 0.7×(1−0.65) = 0.245.

Then N = λT = 38×0.245 = 9.31, which is close to the value 9.7 from the simulation.



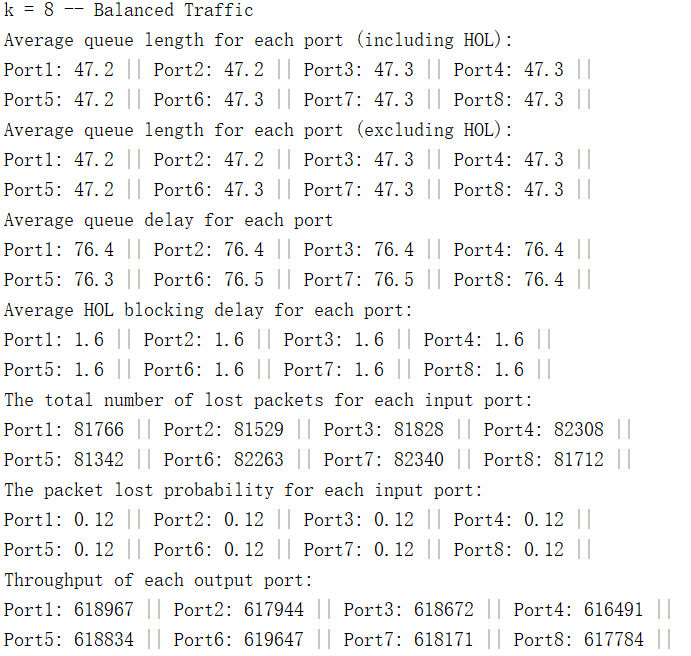
In this case, T is about 25, and the λ = Parrival ×(1−Plost) = 0.7×(1−0.46) = 0.378.

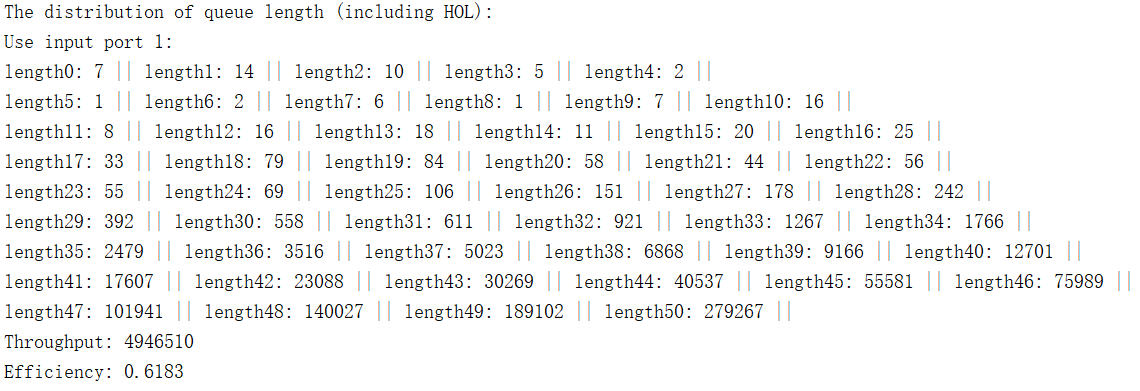
Then N = λT = 25×0.378 = 9.45, which is close to the value 9.3 from the simulation.



In this case, T is about 7.6, and the λ = Parrival ×(1−Plost) = 0.7×(1−0.12) = 0.616.

Then N = λT = 12×0.616 = 7.392, which is close to the value 7.6 from the simulation.





In this case, bufferSize is changed to 50.

T is about 76, and the λ = Parrival ×(1−Plost) = 0.8×(1−0.22) = 0.624.

Then N = λT = 76×0.624 = 47.6, which is close to the value 47 from the simulation.

***5. Source Code***

**Code 1**

N = 8;

oneSecond = 10^6; % there are 10^6 cycles per second

p\_arrival = 0.7;

bufferSize = 10; %can be changed to 50

for i = 2 : N

p = 1 / i;

HOL = zeros(1, N);

queue = zeros(1, N);

accum = zeros(1, N);

pps = zeros(1, N); % initialize pps (packets per second)

cycle = 1;

% initialization

totalLength\_hol = zeros(1, N); % the accumulative queue length of each port considering HOL

totalLength = zeros(1, N); % the accumulative queue length of each port without considering HOL

lostPackets = zeros(1, N); % the number of lost packets for each input port

queueDistribution = zeros(N, bufferSize + 1); % distribution of queue length (from 0 to buffer size)

lostProbability = zeros(1, N); % the probability that the packet is lost

arrive = zeros(1, N); % the total number arrives at each port

HOLNumber = zeros(1, N); % record the number of packets going out from the input port j

while cycle <= oneSecond

for j = 1 : N

% the new packet arrives at input port j

if rand() < p\_arrival

arrive(j) = arrive(j) + 1;

queue(j) = queue(j) + 1;

if queue(j) > bufferSize

queue(j) = bufferSize;

lostPackets(1, j) = lostPackets(1, j) + 1;

end

% decide the destination port of the new HOL

if queue(j) == 1

HOL(j) = nextPacket(N, p);

accum(HOL(j)) = accum(HOL(j)) + 1;

end

end

totalLength\_hol(j) = totalLength\_hol(j) + queue(j);

if queue(j) > 1

totalLength(j) = totalLength(j) + queue(j);

end

queueDistribution(j, queue(j) + 1) = ...

queueDistribution(j, queue(j) + 1) + 1;

end

random = zeros(1, N);

for j = 1 : N

if HOL(j) ~= 0

random(HOL(j)) = randi(accum(HOL(j)));

end

end

for j = 1 : N

if HOL(j) ~= 0

random(HOL(j)) = random(HOL(j)) - 1;

if random(HOL(j)) == 0

HOLNumber(j) = HOLNumber(j) + 1;

pps(HOL(j)) = pps(HOL(j)) + 1;

accum(HOL(j)) = accum(HOL(j)) - 1;

queue(j) = queue(j) - 1;

if queue(j) > 0

HOL(j) = nextPacket(N, p);

accum(HOL(j)) = accum(HOL(j)) + 1;

else

HOL(j) = 0;

end

end

end

end

cycle = cycle + 1;

end

throughput = sum(pps);

fprintf('k = %d -- ', i);

if i ~= N

fprintf('Hot-spot Traffic\n');

else

fprintf('Balanced Traffic\n')

end

averageLength = totalLength\_hol ./ oneSecond;

fprintf("Average queue length for each port (including HOL):\n");

for j = 1 : N

if mod(j, 5) == 0

fprintf("\n");

end

fprintf("Port%d: %.1f || ", j, averageLength(j));

end

fprintf('\n');

averageLength\_queue = totalLength ./ oneSecond;

fprintf("Average queue length for each port (excluding HOL):\n");

for j = 1 : N

if mod(j, 5) == 0

fprintf("\n");

end

fprintf("Port%d: %.1f || ", j, averageLength\_queue(j));

end

fprintf('\n');

queueDelay = averageLength\_queue .\* (oneSecond ./ HOLNumber);

fprintf("Average queue delay for each port\n");

for j = 1 : N

if mod(j, 5) == 0

fprintf("\n");

end

fprintf("Port%d: %.1f || ", j, queueDelay(j));

end

fprintf('\n');

fprintf("Average HOL blocking delay for each port:\n");

HOLDelay = oneSecond ./ HOLNumber;

for j = 1 : N

if mod(j, 5) == 0

fprintf("\n");

end

fprintf("Port%d: %.1f || ", j, HOLDelay(j));

end

fprintf('\n');

fprintf("The total number of lost packets for each input port:\n");

for j = 1 : N

if mod(j, 5) == 0

fprintf("\n");

end

fprintf('Port%d: %d || ', j, lostPackets(j));

end

fprintf('\n');

fprintf("The packet lost probability for each input port:\n");

for j = 1 : N

if mod(j, 5) == 0

fprintf("\n");

end

fprintf('Port%d: %.2f || ', j, lostPackets(j) / arrive(j));

end

fprintf('\n');

fprintf('Throughput of each output port: \n')

for j = 1 : N

if mod(j, 5) == 0

fprintf('\n');

end

fprintf('Port%d: %d || ', j, round(pps(j)));

end

fprintf('\n');

fprintf("The distribution of queue length (including HOL):\n");

fprintf("Use input port 1:\n");

for j = 1 : bufferSize + 1

if mod(j, 6) == 0

fprintf('\n');

end

fprintf("length%d: %d || ", j - 1, queueDistribution(1, j));

end

fprintf('\n');

fprintf('Throughput: %d\n', round(throughput));

efficiency = throughput / (N \* oneSecond);

fprintf('Efficiency: %.4f\n', efficiency);

fprintf('\n');

end

**Code 2**

function port = nextPacket(N, p)

if rand() < p

port = 1;

else

port = randi(N-1) + 1;

end

end