Project 4 Switch Performance and HOL Blocking

EE 511 – Section Thursday 9 am

Name: Junquan Yu

Student ID #: 3372029142

1. Problem Statement

As discussed in class, you are study the operation of an *NxN* switch under heavy traffic, i.e. there are always packets on the input ports. In particular, the HOL (head of the line) slot is always full. The packet in the HOL position at any input of the *N* input ports is addressed to output port $j : j = 1,\ldots, N$ with probability α_j and $\sum_{j=1}^{N} \alpha_j$. Packets can be delivered from inputs to outputs in one clock cycle and the clock rate is 106 cycles per second. If there is more than 1 packet destined to a specific port, only one of them can be delivered in the current slot, the others will remain in the HOL position on the input side. This will reduce switch throughput and is known as HOL-Blocking. Estimate the rate in pps (pps = packets per second) at which packets are delivered to each output port and the overall throughput of the switch (also in pps.)

N=2

We are interested in finding the switch throughput for a range of values for

$\alpha_1 = \{0,0.1,0.2,0.3,0.4,0.5\}$ and recall that $\alpha_2 = 1 - \alpha_1$

Method 1: Build the transition probability matrix (see class notes) and solve the Markov chain numerically to find the limit distribution. From this limit distribution, calculate throughput for each output port and the overall switch performance.

Method 2: Build a simulation model that simulates the operation of the switch. In each slot, one or two packets will be delivered to the output. The packets that are transmitted are replaced in the HOL positions with new packets with destination ports generated randomly according to α_j.

For N =4 and N =8, appropriately generalize the simulation program for these cases (the numerical solution to the balance equations is not required). The number of packets delivered to any particular output in one slot is either 1 or 0. When there are multiple requests for the same output, one of them is selected to be delivered and the others remain in the HOL position on the input side. Simulate for 1) balanced traffic (α_j=1/N $\forall j$ and 2) Hot-spot traffic $\alpha_1 = 1/k$, $\alpha_j = (\frac{1}{N-1})(\frac{K-1}{K})$ for $j \neq 1$. Look at the cases k=2, 3, ..., N (the case of k=N, reverts to being balanced traffic.)

2. Theoretical Exploration or Analysis

Consider the system in state X(t) at time t, where X(t) is a random variable in this problem, we consider discrete time with unit of a single time slot in other word a cycle, and time index is an integer. In order to describe the future of the system state X(t+1), we only need to know the state X(t), which is an important property of Markov chain.

Let X_n be a discrete-time integer-valued Markov chain that starts at n=0 with pmf:

$p_j(0) = P[X_0 = j]$ j=0, 1, 2, …

We will assume X_n that takes on values from a countable set of integers, usually {0, 1, 2, …}. We say that the Markov chain is finite state if takes on values from a finite set.

The joint pmf for the first n+1 values of the process is:

$$P[X_n = i_n, \ldots, X_0 = i_0] = P[X_n = i_n \mid X_{n-1} = i_{n-1}] \ldots P[X_1 = i_1 \mid X_0 = i_0] P[X_0 = i_0]$$

Thus the joint pmf for a particular sequence is simply the product of the probability for the initial state and the probabilities for the subsequent one-step state transitions.

We will assume that the one-step state transition probabilities are fixed and do not change with time, that is,

$$P[X_{n+1} = j \mid X_n = i] = p_{ij} \qquad \text{for all } n$$

X_n is said to have homogeneous transition probabilities. The joint pmf for $X_n, \ldots, X_0$ is then given by:

$$P[X_n = i_n, \ldots, X_0 = i_0] = p_{i_{n-1}, i_n} \ldots p_{i_0, i_1} p_{i_0}(0)$$

Thus X_n is completely specified by the initial pmf $p_i(0)$ and the matrix of one-step transition probabilities P:

$$P = \begin{bmatrix} p_{00} & p_{01} & p_{02} & \cdots \\ p_{10} & p_{11} & p_{12} & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ p_{i0} & p_{i1} & \cdots & \cdots \end{bmatrix}$$

We will call P the transition probability matrix**,** and that each row of P must add to one since:

$$1 = \sum_j P[X_{n+1} = j \mid X_n = i] = \sum_j p_{ij}$$

If the Markov chain is finite state, then the matrix P will be an n x n nonnegative square with rows that add up to 1.

Through the above mathematical derivation, we can easily get the transfer probability as follows:

When α_1=0:

P	(0,0)	(0,1)	(1,0)	(1,1)
(0,0)	0	0	0	1
(0,1)	0	0	0	1
(1,0)	0	0	0	1
(1,1)	0	0	0	1

When α_1=0.1:

P	(1,1)	(1,2)	(2,1)	(2,2)
(1,1)	0.1	0.45	0.45	0
(1,2)	0.01	0.09	0.09	0.81
(2,1)	0.01	0.09	0.09	0.81
(2,2)	0	0.45	0.45	0.1

When α_1=0.3:

P	(1,1)	(1,2)	(2,1)	(2,2)
(1,1)	0.3	0.35	0.35	0
(1,2)	0.09	0.21	0.21	0.49
(2,1)	0.09	0.21	0.21	0.49
(2,2)	0	0.35	0.35	0.3

When α_1=0.4:

P	(1,1)	(1,2)	(2,1)	(2,2)
(1,1)	0.4	0.3	0.3	0

(1,2)	0.16	0.24	0.24	0.36
(2,1)	0.16	0.24	0.24	0.36
(2,2)	0	0.3	0.3	0.4

When α_1=0.5:

P	(1,1)	(1,2)	(2,1)	(2,2)
(1,1)	0.5	0.25	0.25	0
(1,2)	0.25	0.25	0.25	0.25
(2,1)	0.25	0.25	0.25	0.25
(2,2)	0	0.25	0.25	0.5

After these transition matrixes were generated, we can let the program use these parameters to calculate and get limiting distribution.

3. Simulation Methodology

For the problem 1:

According to the first method, we should use transition matrix to solve the problem numerically while the second one is based on the simulation. Just assuming the state (1,1) is the first state, because the final result does not vary with the choice of the first state in the Markov Stochastic Process. If the former state is (1,1) or (1,2), only 1 packet can be allowed to be sent, but when the former state is (1,2) or (2,1), it means the system can send 2 packets. That is to say, only 1 or 2 packets can be sent in one cycle in this case. Then the switched process is simulated, we decide how many packets can be send in current cycle on the basis of the former state. Once the old packet being sent, I used the built-in function rand( ) to determine the destination of the new packets with the provided probability α_j.

For the problem 2:

For the circumstances where N=4 or 8, I just derived the result from the same simulation method. Like I did in the problem 1, I just started with state (1,1) again, for I know that the choice of the first state has nothing to do with the final result from the theoretical analysis. Then I decided the total number of packets sent in a cycle and thus determined the destination of the new packets with the parameters α_j, which is the probability for new packet to be switched to the port j, and the built-in function rand( ).

4. Experiments and Results

For the problem 1:

The problem requires me to the simulate for α_1=0, α_1=0.1, α_1=0.2, α_1=0.3, α_1 =0.4, α_1=0.5, so I run the program for one time for each of these values. The results are as follows:

a) Method 1:

α_1	The throughput for output port 1 (pps)	The throughput for output port 2 (pps)	The overall throughput (pps)
0	0	10^5	10^6
0.1	1.1098 x 10^5	9.9878 x 10^5	1.1098 x 10^6
0.2	2.4706 x 10^5	9.8824 x 10^5	1.2353 x 10^6

0.3	4.0862×10^5	9.5345×10^5	1.3621×10^6
0.4	5.8462×10^5	8.7692×10^5	1.4615×10^6
0.5	7.5×10^5	7.5×10^5	1.5×10^6

b) Method 2:

α_1	The throughput for output port 1 (pps)	The throughput for output port 2 (pps)	The overall throughput (pps)
0	0	10^6	10^6
0.1	1.1149×10^5	9.9882×10^5	1.1103×10^6
0.2	2.4649×10^5	9.8848×10^5	1.2350×10^6
0.3	4.0977×10^5	9.5332×10^5	1.3631×10^6
0.4	5.8514×10^5	8.7567×10^5	1.4608×10^6
0.5	7.5015×10^5	7.5009×10^5	1.5002×10^6

It can be seen from the above two tables that result from the method 1 are almost the same as the one from the method 2. That is to say, the simulated value matches well with the theoretical value. As the value of α_1 get larger and larger, both the throughput for the single output port and the overall throughput increase accordingly. When α_1 takes on value of 0.5, the overall throughput reaches its maximum as well, which is about $1.5\text{x}10^6$ pps.

For the problem 2:

a) Balanced traffic

After running the program, I input N=4 and N=8 successively as the total number of the input ports and it returns the result as follows:

Throughput (pps) \ N	4	8
The throughput for port 1	6.5643×10^5	6.1929×10^5
The throughput for port 2	6.5474×10^5	6.1821×10^5
The throughput for port 3	6.5539×10^5	6.1865×10^5
The throughput for port 4	6.5352×10^5	6.1890×10^5
The throughput for port 5	NA	6.1885×10^5
The throughput for port 6	NA	6.1840×10^5
The throughput for port 7	NA	6.1851×10^5
The throughput for port 8	NA	6.1700×10^5
The overall throughput	2.6201×10^6	4.9478×10^6

As can be seen from above table that the throughput for each output port is almost at the same level in the balanced traffic. Moreover, with the value taken on by N from 4 to 8, the throughput for every single output port slightly decreases while the overall throughput significantly boosts.

b) Hot-spot traffic

After running the program, when N=4, I simulated for three times with the value taken on by k from 2 to 4; when N=8, I simulated for seven times with the value taken on by k from 2 all the way to 8. The results are as follows:

When N=4:

Throughput \ k	2	3	4
The overall	1.9776×10^6	2.5304×10^6	2.6209×10^6

throughput			
The throughput for port 1	9.8861×10^5	8.4413×10^5	6.5434×10^5
The throughput for port 2	3.2965×10^5	5.6140×10^5	6.5608×10^5
The throughput for port 3	3.2998×10^5	5.6189×10^5	6.5520×10^5
The throughput for port 4	3.2932×10^5	5.6296×10^5	6.5529×10^5

When N=8:

Throughput\k	2	3	4	5	6	7	8
The overall throughput	2.0019×10^6	3.0024×10^6	3.9401×10^6	4.5742×10^6	4.8427×10^6	4.9311×10^6	4.9459×10^6
The throughput for port 1	10^6	9.9967×10^5	9.8447×10^5	9.1412×10^5	8.0749×10^5	7.0536×10^5	6.1991×10^5
The throughput for port 2	1.4344×10^5	2.8581×10^5	4.2068×10^5	5.2328×10^5	5.7653×10^5	6.0496×10^5	6.1779×10^5
The throughput for port 3	1.4311×10^5	2.8636×10^5	4.2273×10^5	5.2298×10^5	5.7552×10^5	6.0385×10^5	6.1805×10^5
The throughput for port 4	1.4287×10^5	2.8642×10^5	4.2240×10^5	5.2385×10^5	5.7564×10^5	6.0314×10^5	6.1907×10^5
The throughput for port 5	1.4321×10^5	2.8570×10^5	4.2177×10^5	5.2219×10^5	5.7711×10^5	6.0357×10^5	6.1801×10^5
The throughput for port 6	1.4311×10^5	2.8655×10^5	4.2274×10^5	5.2375×10^5	5.7773×10^5	6.0375×10^5	6.1828×10^5
The throughput	1.4286	2.8534	4.2262	5.2146	5.7711	6.0429	6.1716

for port 7	x 10^5	x 10^5	x 10^5	x 10^5	x 10^5	x 10^5	x 10^5
The throughput for port 8	1.4326 x 10^5	2.8662 x 10^5	4.2266 x 10^5	5.2257 x 10^5	5.7562 x 10^5	6.0223 x 10^5	6.1758 x 10^5

It can be seen from the above tables that in the Hot spot traffic, for the given N (whether it is four or eight), with the increase of the value taken on by k, the throughput for output port 1 decreases while the overall throughput and the throughput for all the other output ports except the port 1 both boost. Moreover, it is worth noting that the throughput for every single output port is almost the same when k=N.

5. References

1. *Alberto Leon-Garcia. (2008). Probability, Statistics, and Random Processes for Electrical Engineering. Upper Saddle River, NJ 07458. Pearson Education, Inc.*

2. *Zhou Sheng, Shiqian Xie, Chengyi Pan. (2008). Probability Theory and Mathematical Statistics. No.4, Dewai Street, Xicheng District, Beijing. Higher Education Press.*

6. Source Code

For the problem 1:

a) Method 1:

```
a1=input('a1=');        %enter the probability that the packet is switched to port 1
a2=1-a1;
p_matrix=[a1,0.5*a2,0.5*a2,0;      %set the transition matrix
         a1^2,a1*a2,a1*a2,a2^2;
         a1^2,a1*a2,a1*a2,a2^2;
         0,0.5*a1,0.5*a1,a2];
state=[1,0,0,0];
limit=1000000;
for it=1:limit                    %set the total number of calculation loop
    state=state*p_matrix;
end
tptotal=(state(1)+state(4)+2*(state(2)+state(3)))*10^6        %compute the throughput for
                                                                 each output port and the
                                                                 overall throughput
tp1=(state(1)+state(2)+state(3))*10^6
```

```
tp2=(state(4)+state(2)+state(3))*10^6
```

b) Method 2:

```
a1=input('a1=');        %enter the probability that the packet is switched to port 1
limit=1000000;           %set the total number of loop
sum=0;                   %initialize the accumulator of packets
singleport=[0,0];
output=[2,2];            %set the first state
for j=1:limit            %the loop begins
   if output(1)==output(2)
      sum=sum+1;         %decide the throughput of the each output port
      singleport(output(1))=singleport(output(1))+1;
      if rand<a1         %decide the destination of the new packet
         output(1)=1;
      else
         output(1)=2;
      end
   else
      sum=sum+2;         %decide the throughput of the each output port
      singleport(1)=singleport(1)+1;
      singleport(2)=singleport(2)+1;
      x=rand(1,2);
      for i=1:2          %decide the destination of the new packet
         if x(i)<a1
            output(i)=1;
         else
            output(i)=2;
         end
      end
   end
end
tptotal=sum              %compute the throughput for each output port and the overall
                          throughput
tp1=singleport(1)
tp2=singleport(2)
```

For the problem 2:

a) Balanced traffic:

```
N=input('Enter total number of input ports N=');   %enter total number of input ports
limit=1000000;                                        %set the total number of loop

singleport=zeros(1,N);
output=zeros(1,N)+1;      %set the first state
```

```
sum=0;                      %initialize the accumulator of packets
for i=1:N                   %define the probability of the each output port
    a(i)=1/N;
end
prob(1)=a(1);               %create the array of the probability
for i=2:N
    prob(i)=a(i)+ prob(i-1);
end
for j=1:limit               %the loop begins
    sum=sum+length(unique(output));        %decide the throughput of the single output port
    k=unique(output);
    for i=1:length(x)
        singleport(x(i))=singleport(x(i))+1;
    end
    for i=1:length(x)       %decide the destination of the new packet
        position=find(output==x(i));
        P(i)=position(1);
        m=1;
        q=rand;
        while q>prob(m)
            m=m+1;
        end
        output(P(i))=m;     %the destination of the replaced packet
    end
end
tptotal=sum             %compute and return the throughput for each output port and the
                           overall throughput
singleport
```

b) Hot-spot traffic:

```
N=input('Enter total number of input ports N=');   %enter total number of input ports
k=input('Enter the parameter k=');                 %specify the parameter k
singleport=zeros(1,N);
limit=1000000;
output=zeros(1,N)+1;     %set the first state
sum=0;                   %initialize the accumulator of packets
a(1)=1/k;                %define the probability of the each output port
for i=2:N
    a(i)=(1/(N-1))*((k-1)/k);
end
prob(1)=a(1);            %create the array of the probability
for i=2:N
    prob(i)=a(i)+ prob(i-1);
end
```

```
for j=1:limit          %the loop begins
   sum=sum+length(unique(output));    %decide the throughput of the single output port
   x=unique(output);
   for i=1:length(x)
      singleport(x(i))=singleport(x(i))+1;
   end
   for i=1:length(x)                 %decide the destination of the new packet
      position=find(output==x(i));
      P(i)=position(1);
      m=1;
      q=rand;
      while q>prob(m)
         m=m+1;
      end
      output(P(i))=m;      %the destination of the replaced packet
   end
end
tptotal=sum               %compute and return the throughput for each output port and the
                           overall throughput
singleport
```