APPLICATION OF NETWORKS
TELECOM 2310 – PROJECT 1

ARUN KUMAR RAJENDRA KUMAR arr105@pitt.edu
SISHIR RAJA CHALASANI sic35@pitt.edu
MALEK ALAHMADI msa65@pitt.edu

INTRODUCTION:

Routers generally store the packets in queue before they can serve them. Once the queue is full, the packets are dropped. Generally this depends on the packet arrival rate and departure rate which in turn depends on λ and μ. The probability of each event needs to be found out for calculating whether the given event is arrival or departure.

The formulas are given below,

$P_{arrival} = \lambda / (\mu + \lambda)$

$P_{departure} = \mu / (\mu + \lambda)$

It is to be noted that each event is a discrete event.

SOFTWARE USED:

The software used here is MATLAB R2016 – academic use. The graphs are plotted through this software. Whenever using MATLAB, remember to save the file in '.m' format.

ALGORITHM:

Initialization phase:

Initialize variables/counters related with the queue status.
pkt in q = 0
pkt dropped = 0

Simulation:

Repeat for x times (that is, the number of events we want to simulate) algorithm 1.

Data: λ (incoming rate), μ (outgoing rate), n (buffer size)

Result: `pkt_in_q`, `pkt_dropped`

```
begin
    y=rand([0,1])
    if y ≤ λ/(μ+λ) then
        if pkt_in_q < n then
            pkt_in_q++
        end
        else
            pkt_dropped++
        end
    end
    else
        if pkt_in_q > 0 then
            pkt_in_q- -
        end
    end
    Write pkt_in_q and pkt_dropped in a file
end
```

PERFORMANCE ANALYSIS:

The above given algorithm needs to be simulated for 1,000,000 times and the corresponding graphs are to be plotted. Here we need to plot packets dropped & packets in queue versus number of events.

CASE 1: Constant Rates

The various values of λ,μ and n are taken. For every combination of those values, the corresponding figure is plotted.

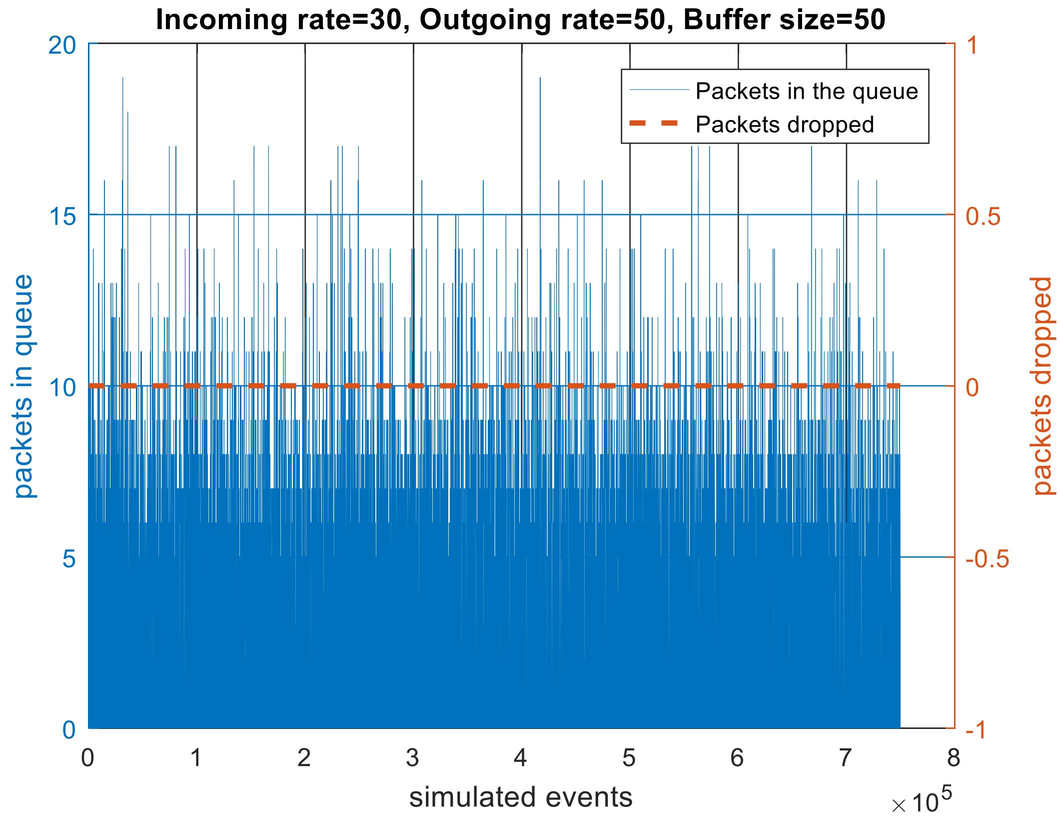
The various combinations are given in the next page,

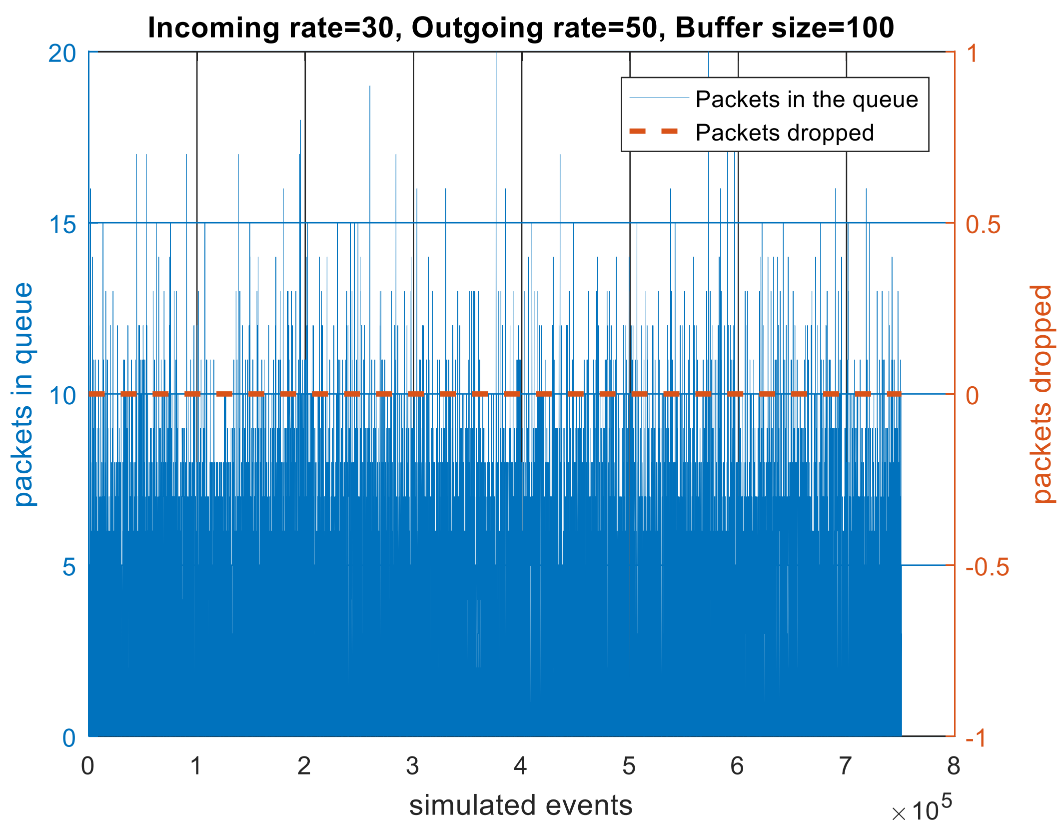
S. No	**λ** (packets/second)	**µ** (packets/second)	**n** (packets)
1)	30	50	50
2)	30	50	100
3)	30	50	150
4)	30	100	50
5)	30	100	100
6)	30	100	150
7)	30	120	50
8)	30	120	100
9)	30	120	150
10)	80	50	50
11)	80	50	100
12)	80	50	150
13)	80	100	50
14)	80	100	100
15)	80	100	150
16)	80	120	50
17)	80	120	100
18)	80	120	150
19)	120	50	50
20)	120	50	100
21)	120	50	150
22)	120	100	50
23)	120	100	100
24)	120	100	150
25)	120	120	50
26)	120	120	100
27)	120	120	150

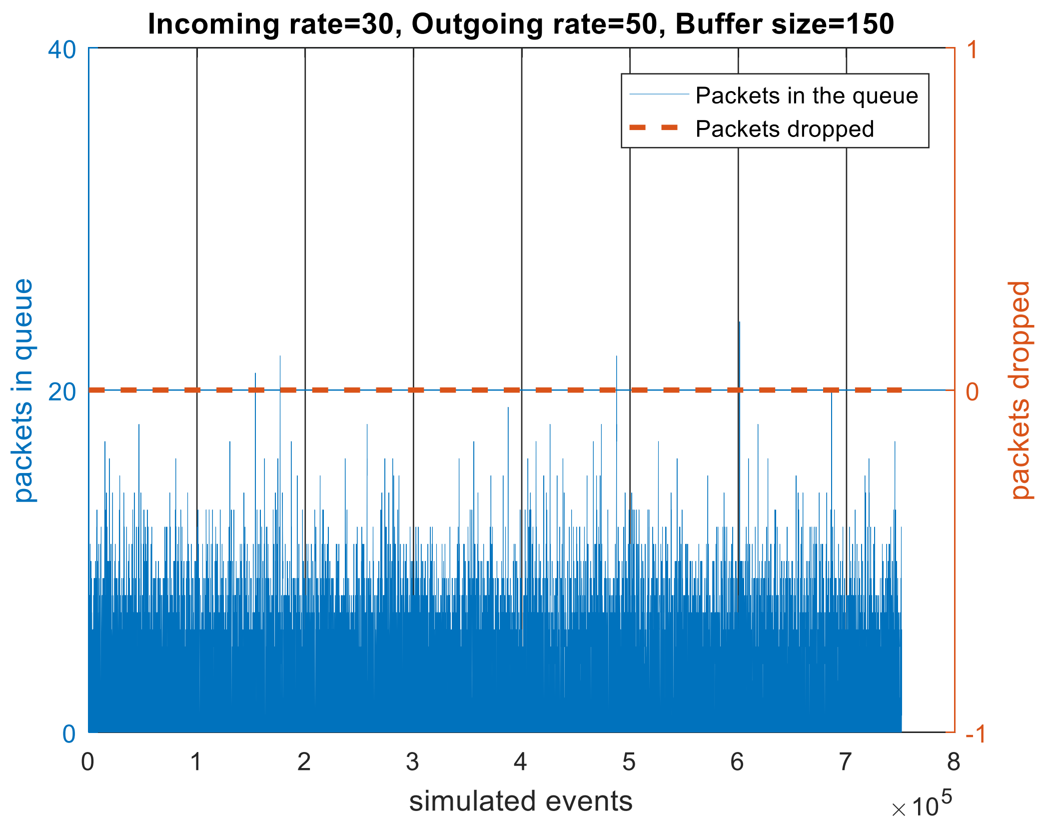
OUTPUT:

Command Window

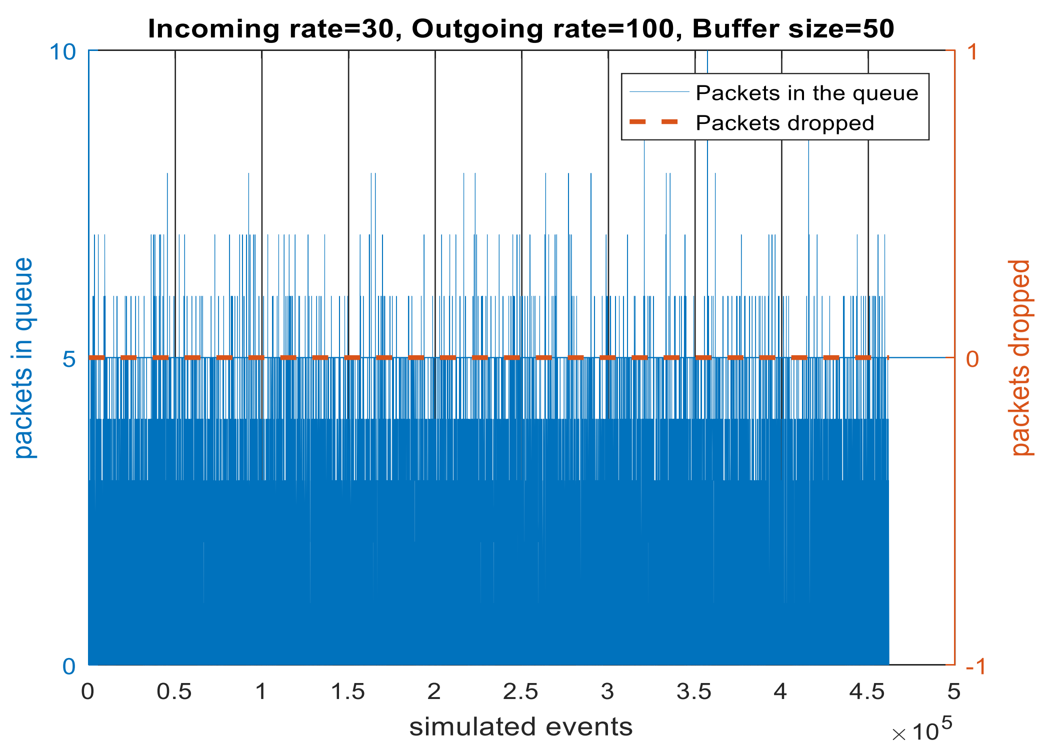
```
Enter the incoming rates: [30 80 120]
Enter the outgoing rates: [50 100 120]
Enter the buffer sizes: [50 100 150]
```


Incoming rate=30, Outgoing rate=50, Buffer size=50
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
×10^5
0
1
2
3
4
5
6
7
8
20
15
10
5
0
1
0.5
0
-0.5
-1

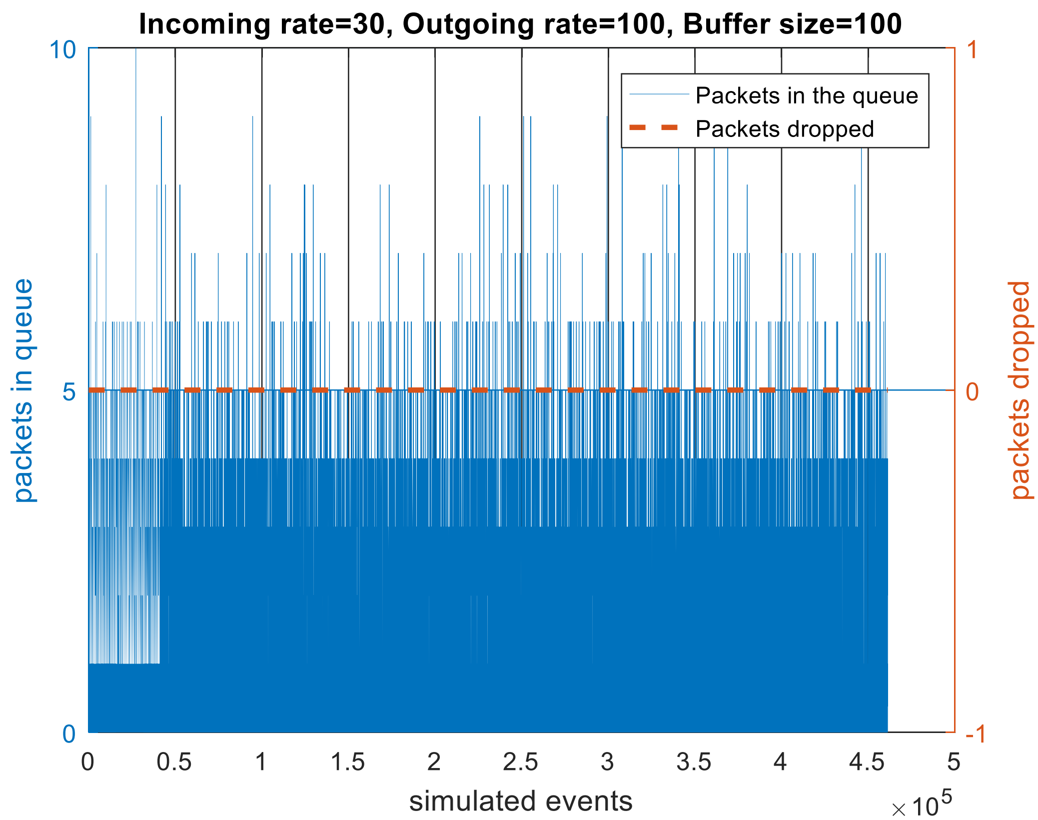

Incoming rate=30, Outgoing rate=50, Buffer size=100
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
×10^5
0
1
2
3
4
5
6
7
8
20
15
10
5
0
1
0.5
0
-0.5
-1

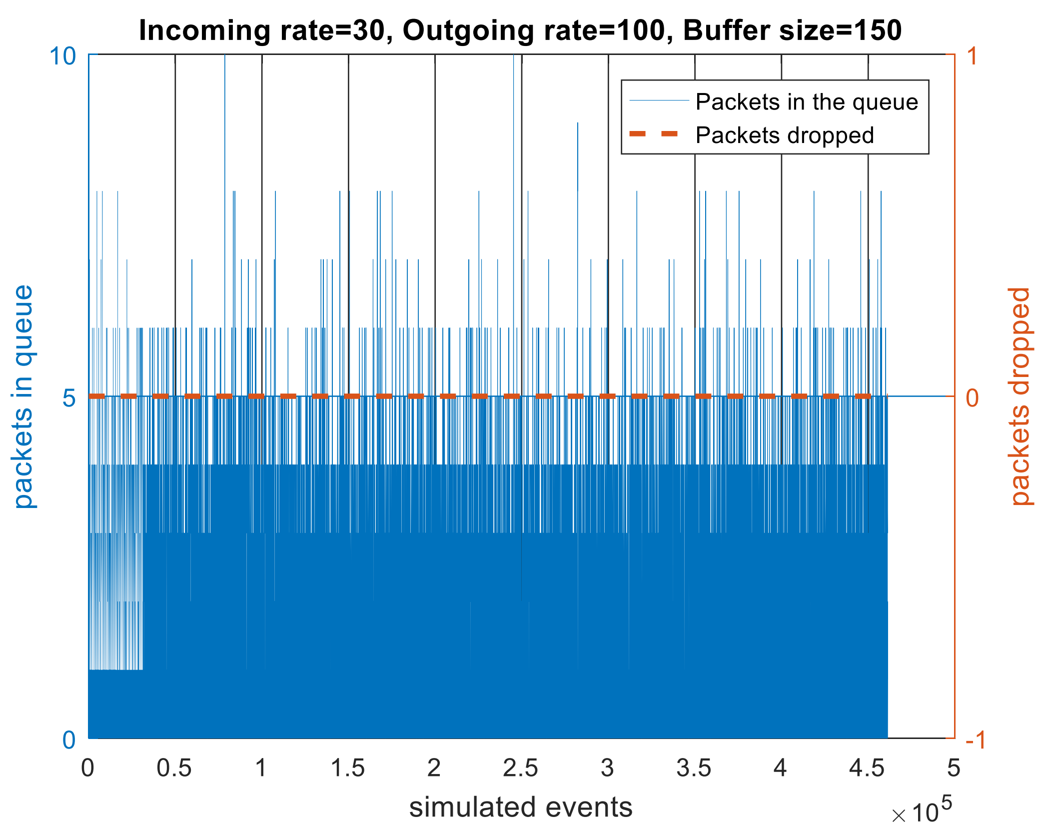


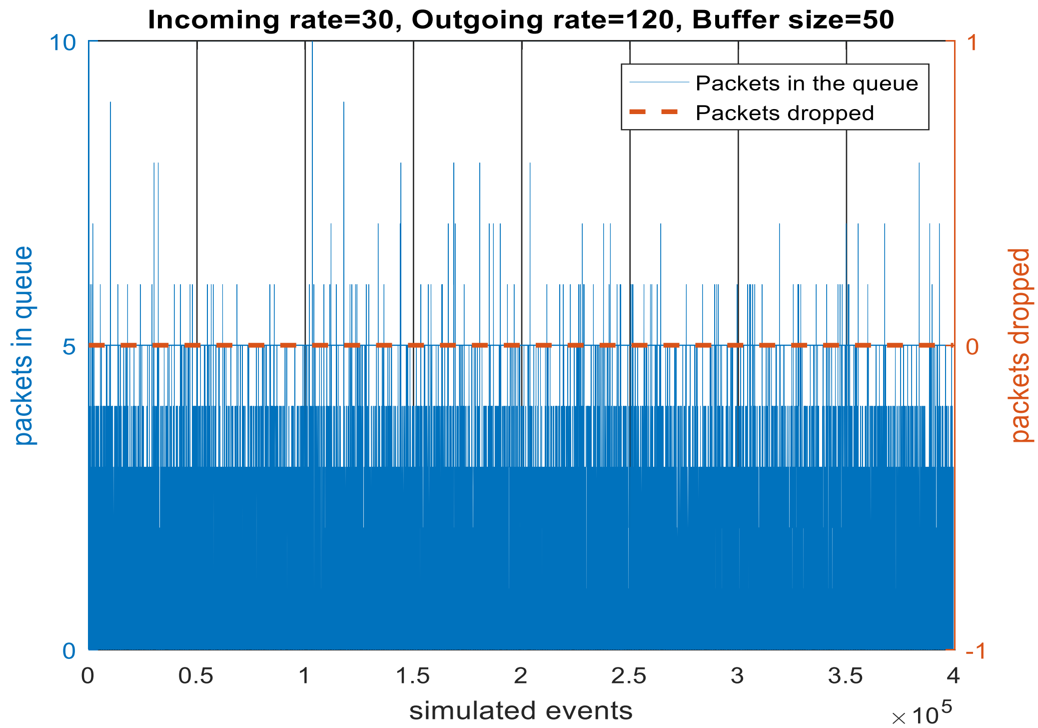
Incoming rate=30, Outgoing rate=50, Buffer size=150
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$



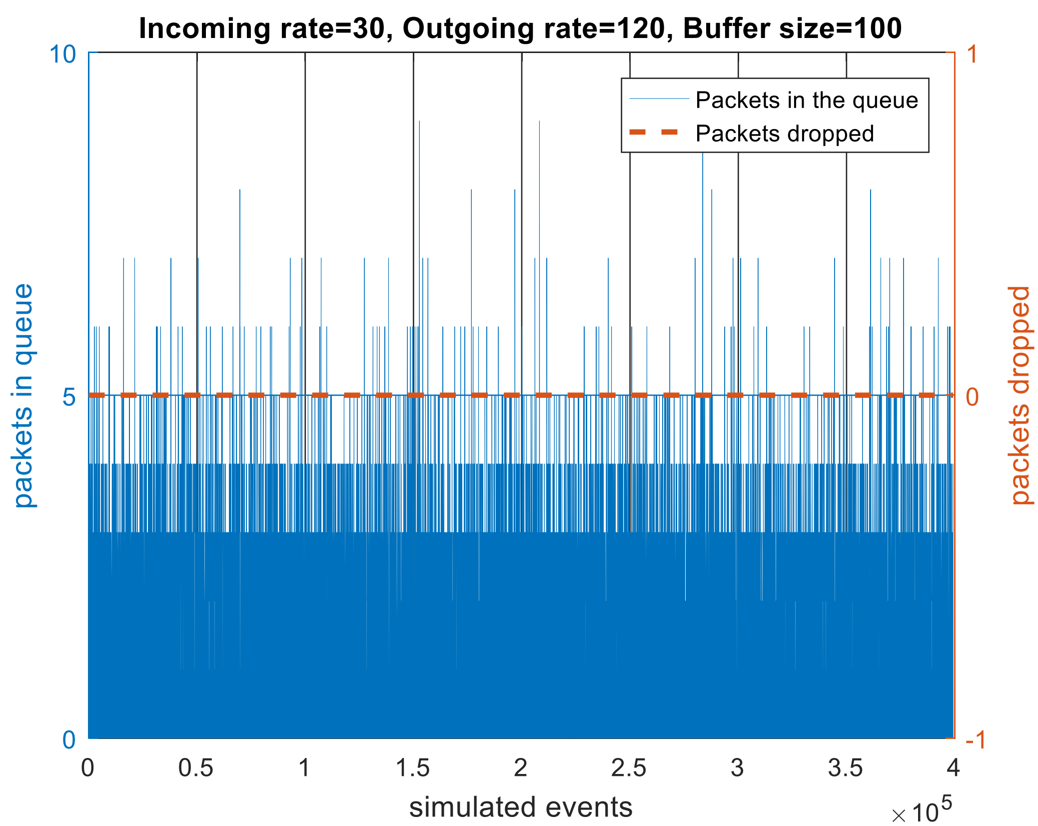
Incoming rate=30, Outgoing rate=100, Buffer size=50
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$


Incoming rate=30, Outgoing rate=100, Buffer size=100
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
$\times 10^5$
10
5
0
1
0
-1

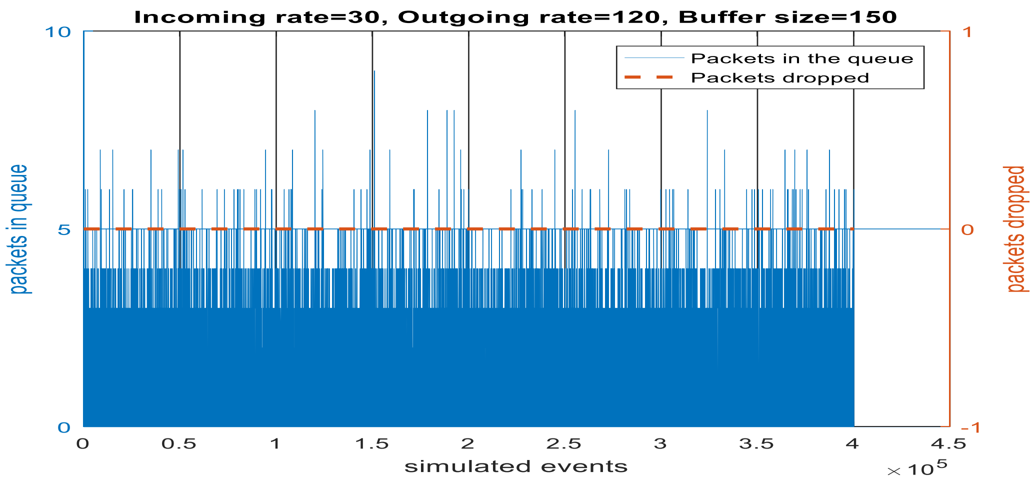

Incoming rate=30, Outgoing rate=100, Buffer size=150
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
$\times 10^5$
10
5
0
1
0
-1

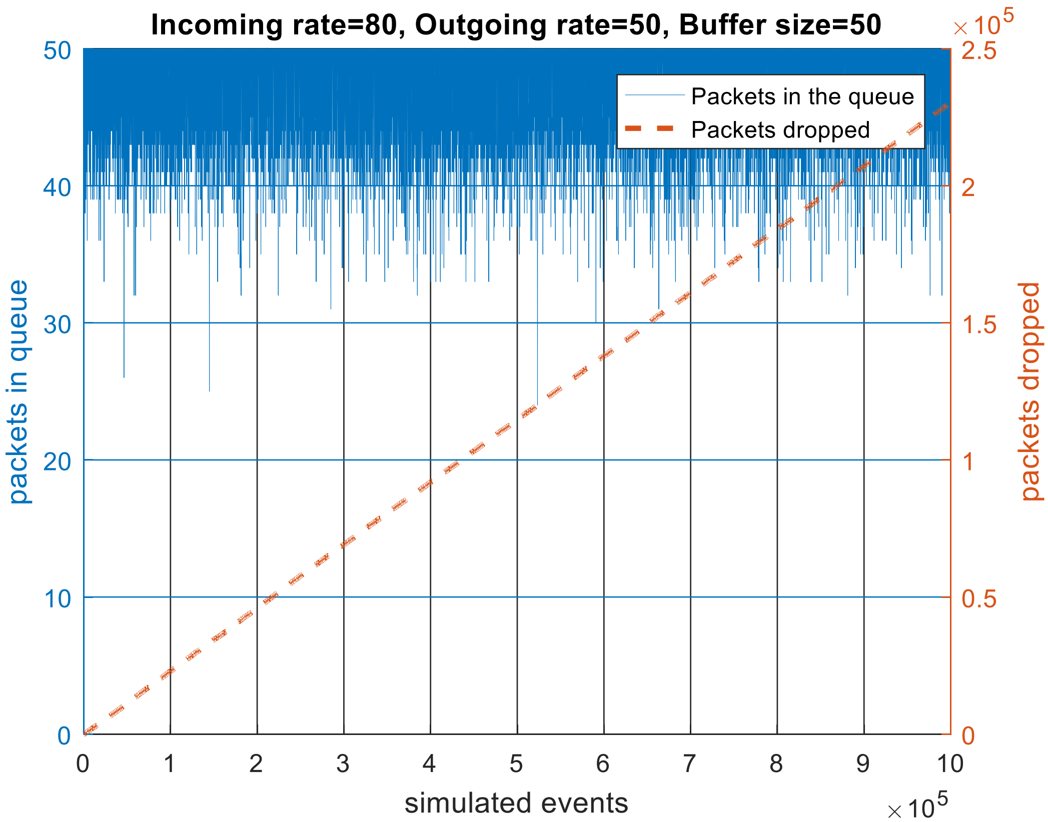


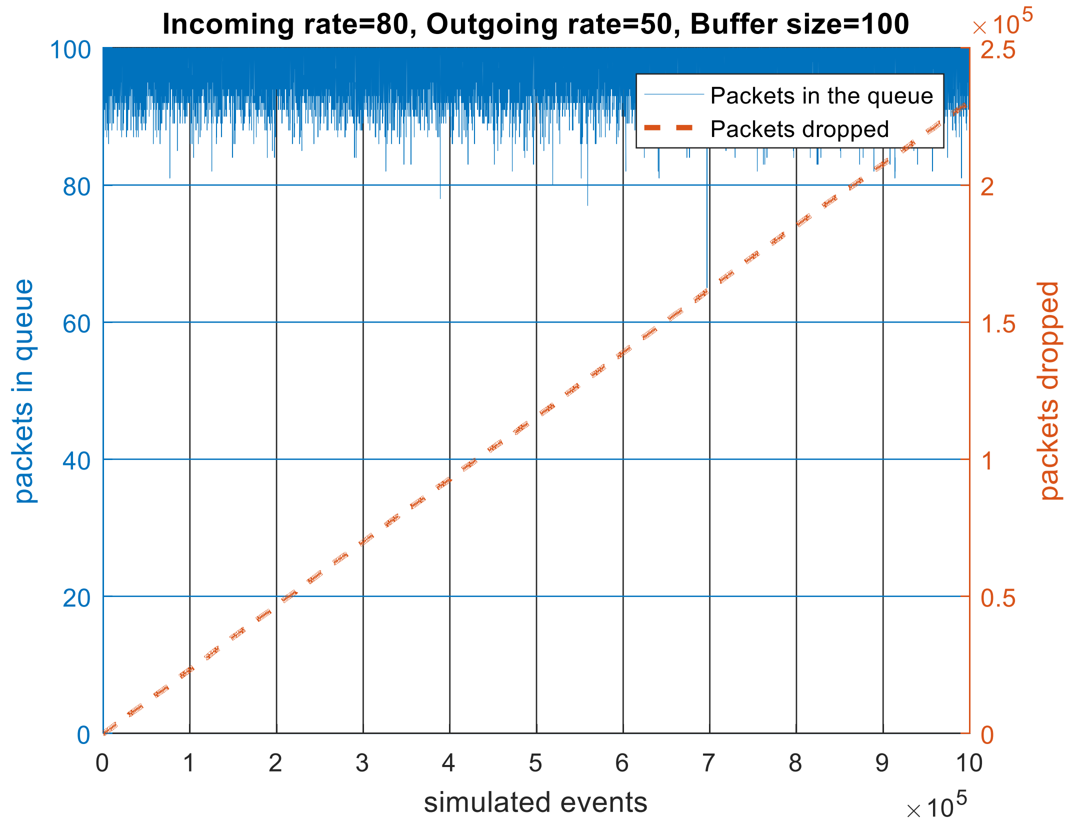
Incoming rate=30, Outgoing rate=120, Buffer size=50
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$
0
0.5
1
1.5
2
2.5
3
3.5
4
5
10
-1

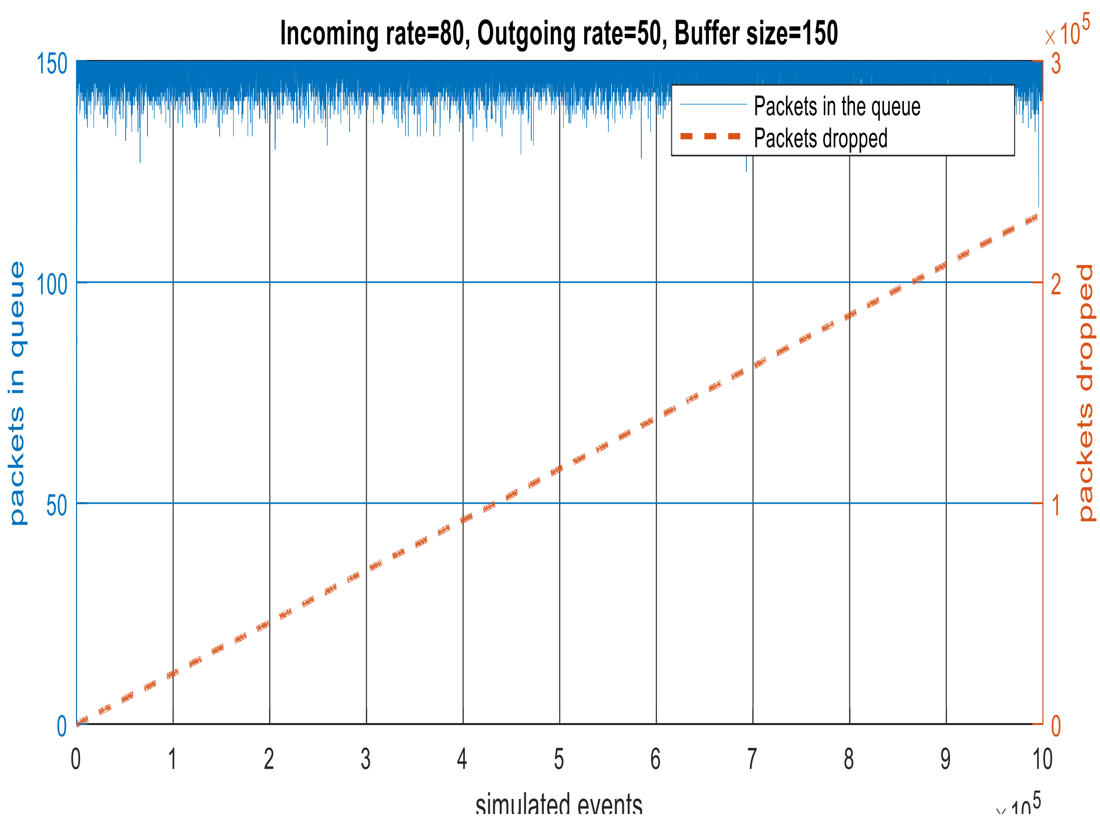


Incoming rate=30, Outgoing rate=120, Buffer size=100
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$
0
0.5
1
1.5
2
2.5
3
3.5
4
5
10
-1

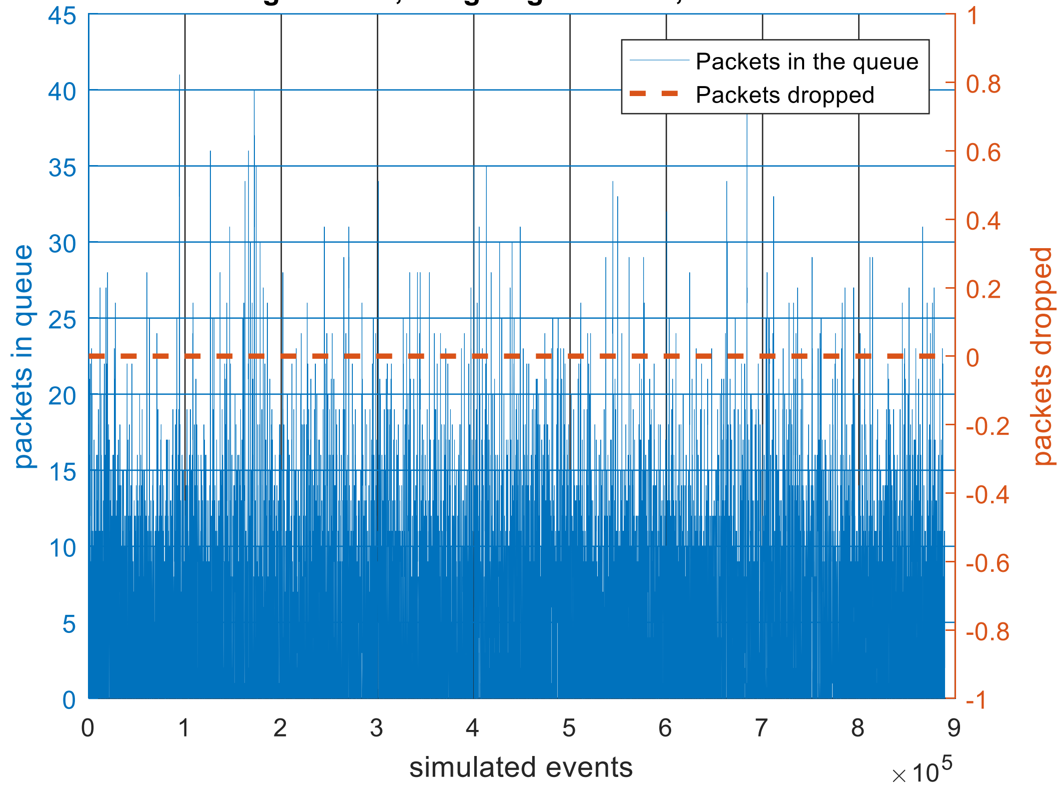

Incoming rate=30, Outgoing rate=120, Buffer size=150
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$


Incoming rate=80, Outgoing rate=50, Buffer size=50
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$
$\times 10^5$

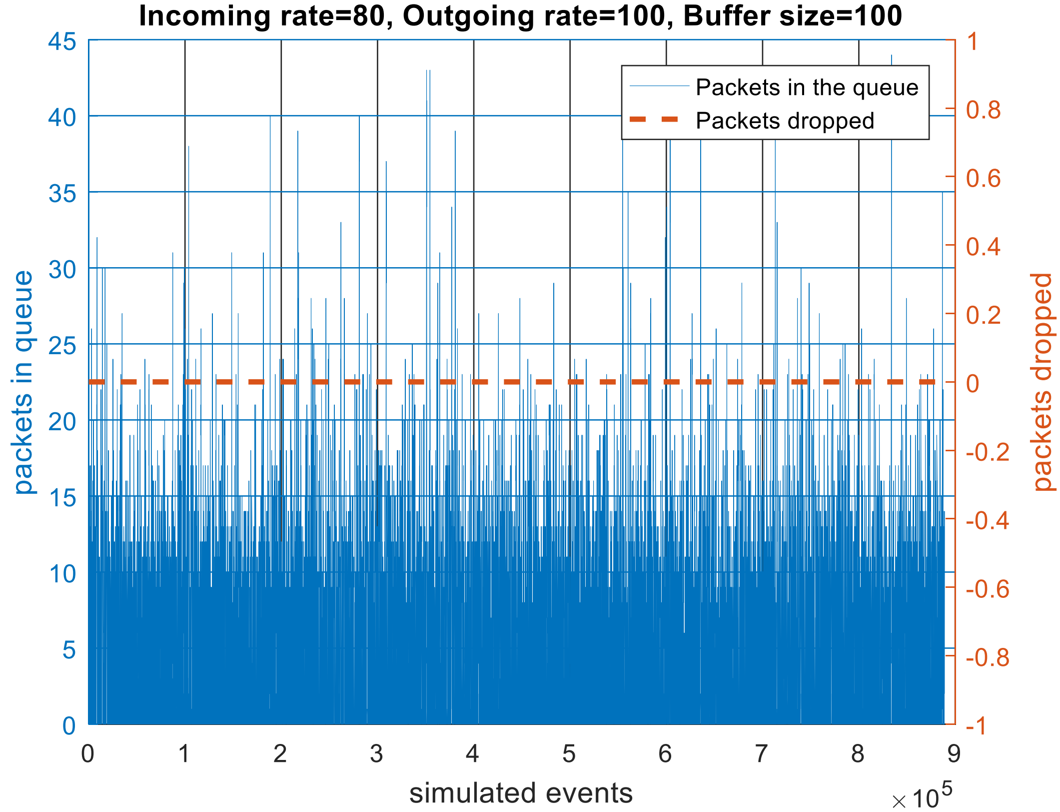

Incoming rate=80, Outgoing rate=50, Buffer size=100
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$
$\times 10^5$

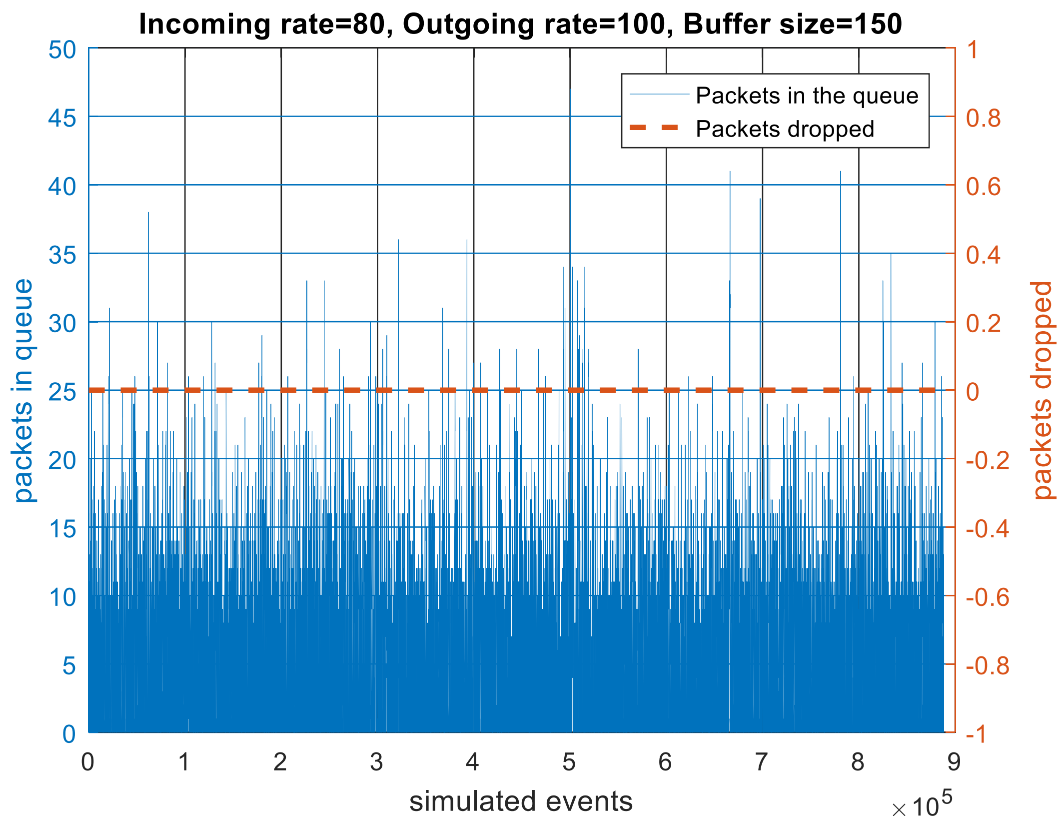

Incoming rate=80, Outgoing rate=50, Buffer size=150
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$
$\times 10^5$

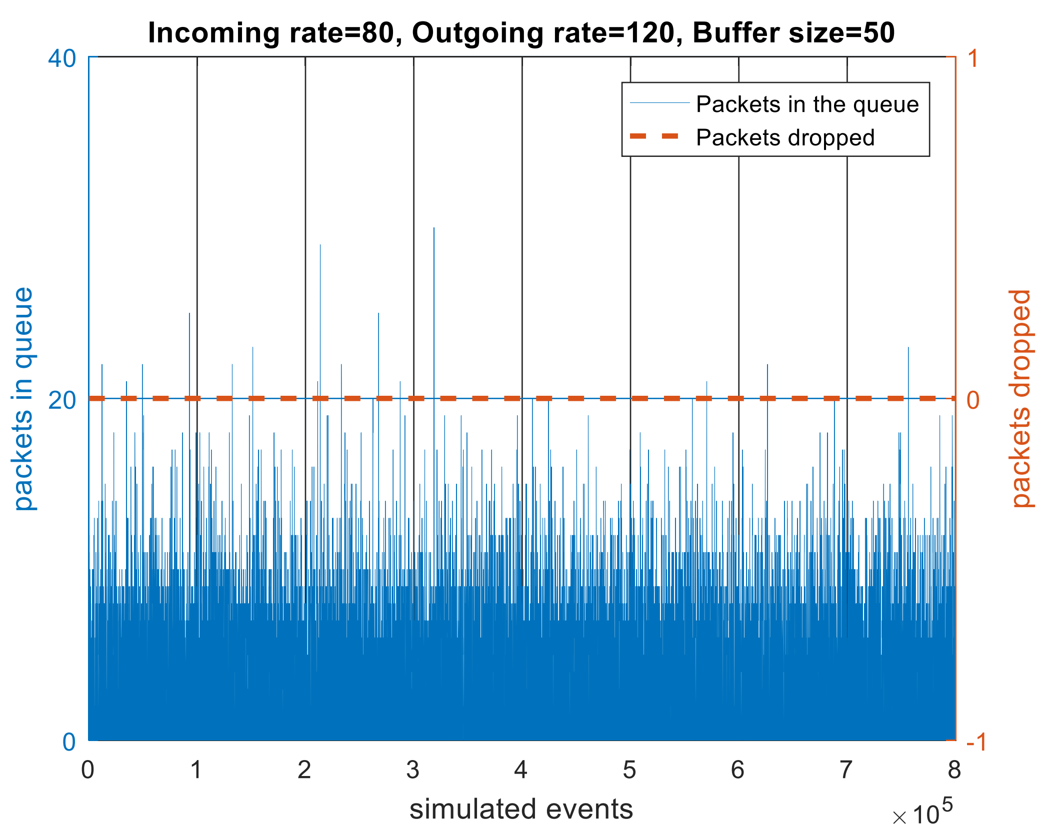
Incoming rate=80, Outgoing rate=100, Buffer size=50

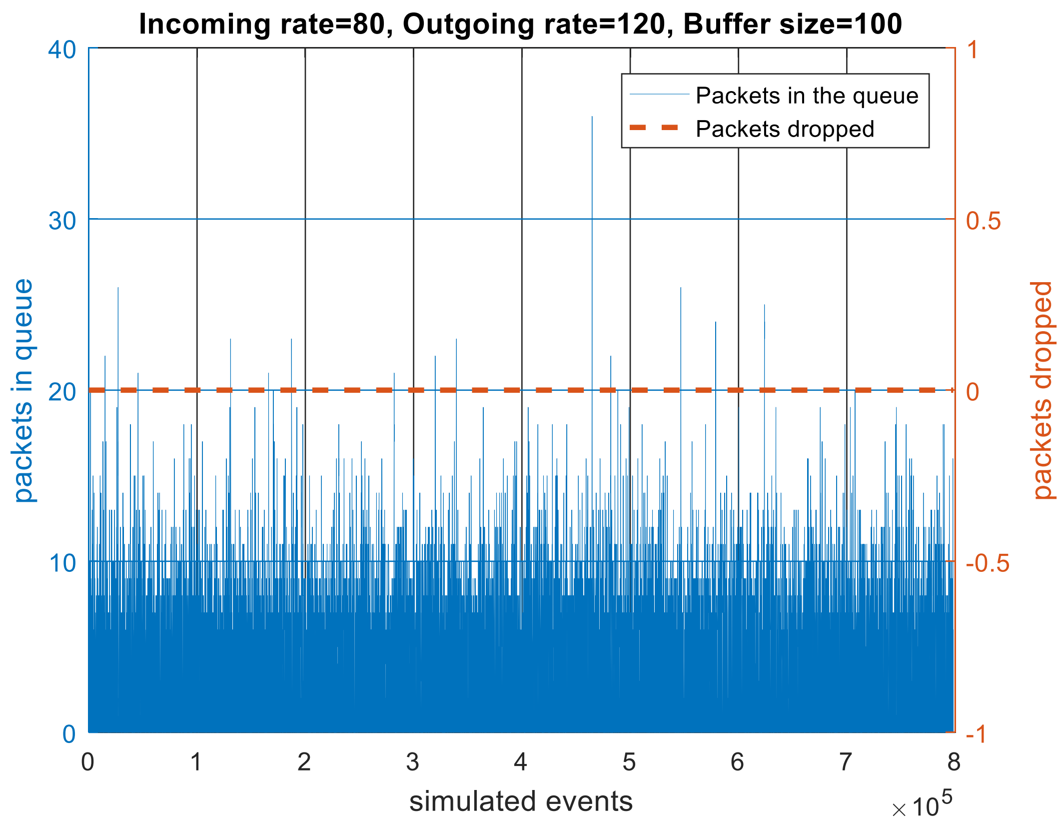


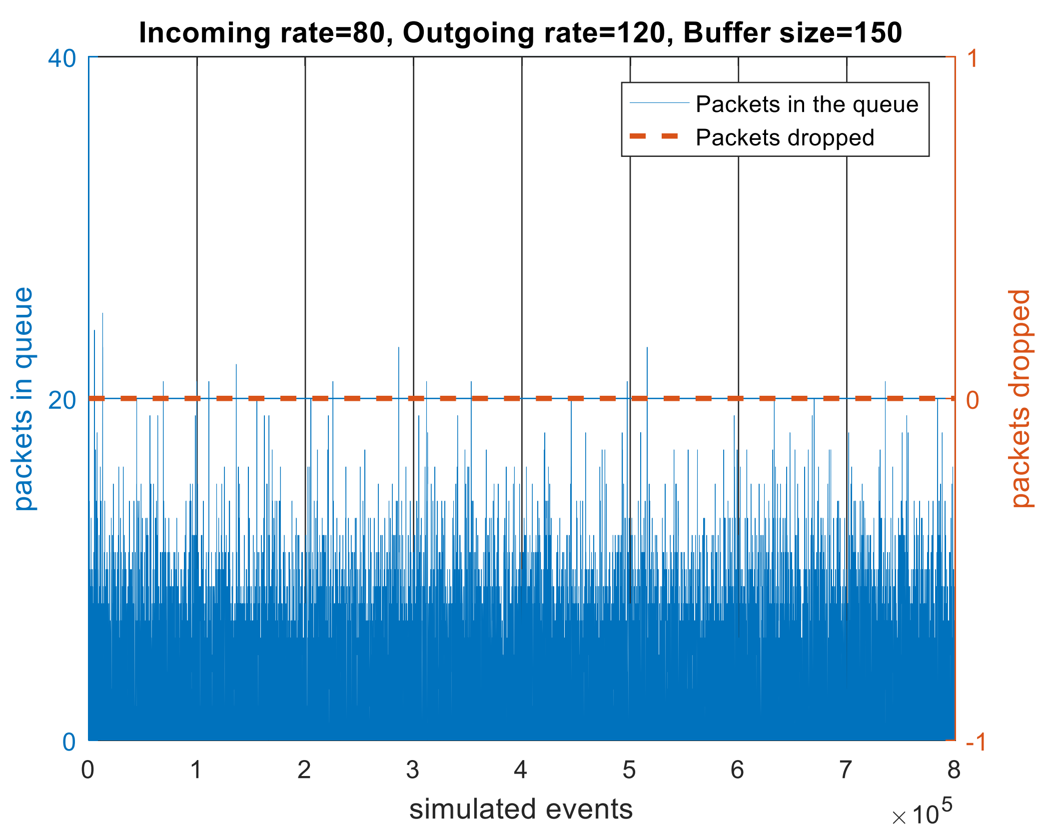
Incoming rate=80, Outgoing rate=100, Buffer size=100

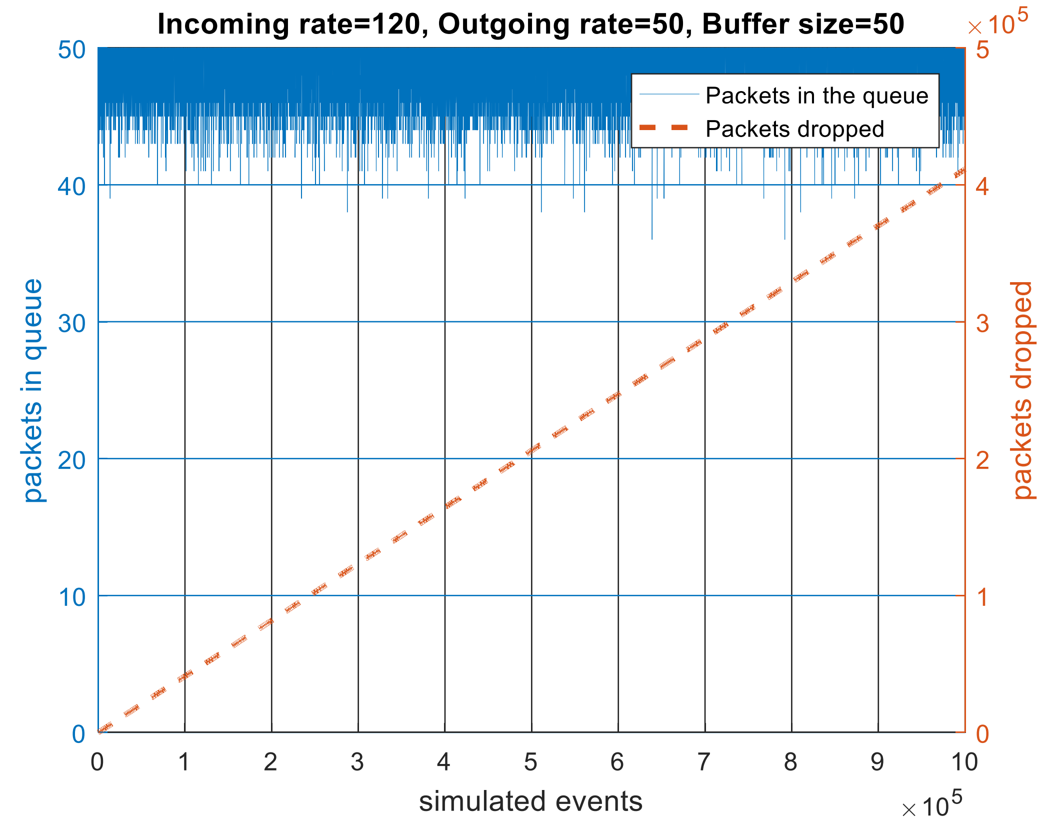



Incoming rate=80, Outgoing rate=100, Buffer size=150
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
×10^5
0
1
2
3
4
5
6
7
8
9
0
5
10
15
20
25
30
35
40
45
50
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

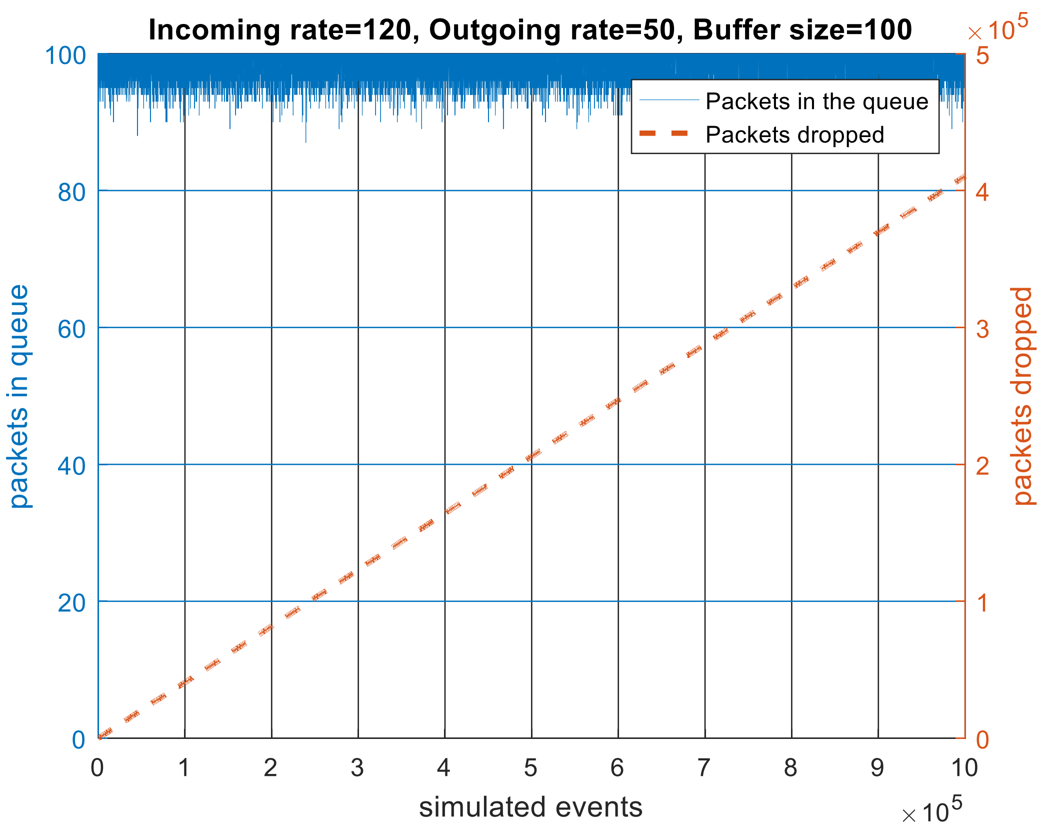

Incoming rate=80, Outgoing rate=120, Buffer size=50
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
×10^5
0
1
2
3
4
5
6
7
8
0
20
40
-1
0
1


Incoming rate=80, Outgoing rate=120, Buffer size=100
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
×10^5
0
10
20
30
40
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6
7
8

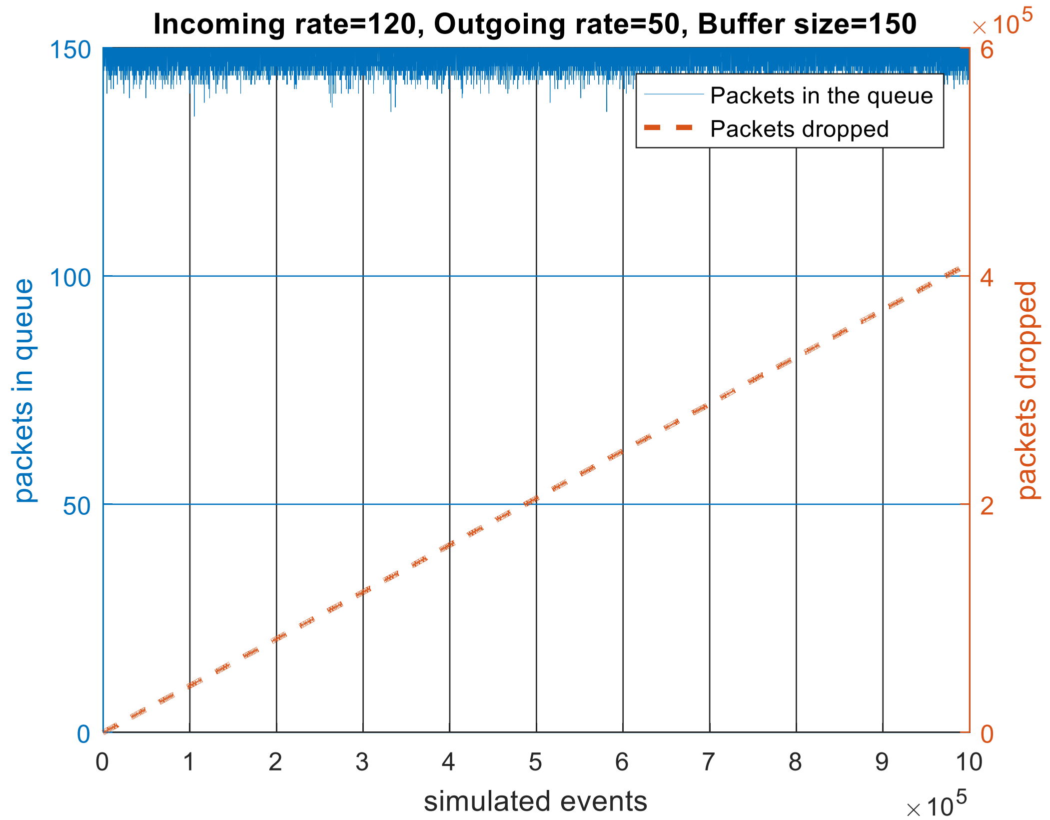

Incoming rate=80, Outgoing rate=120, Buffer size=150
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
×10^5
0
20
40
1
0
-1
0
1
2
3
4
5
6
7
8

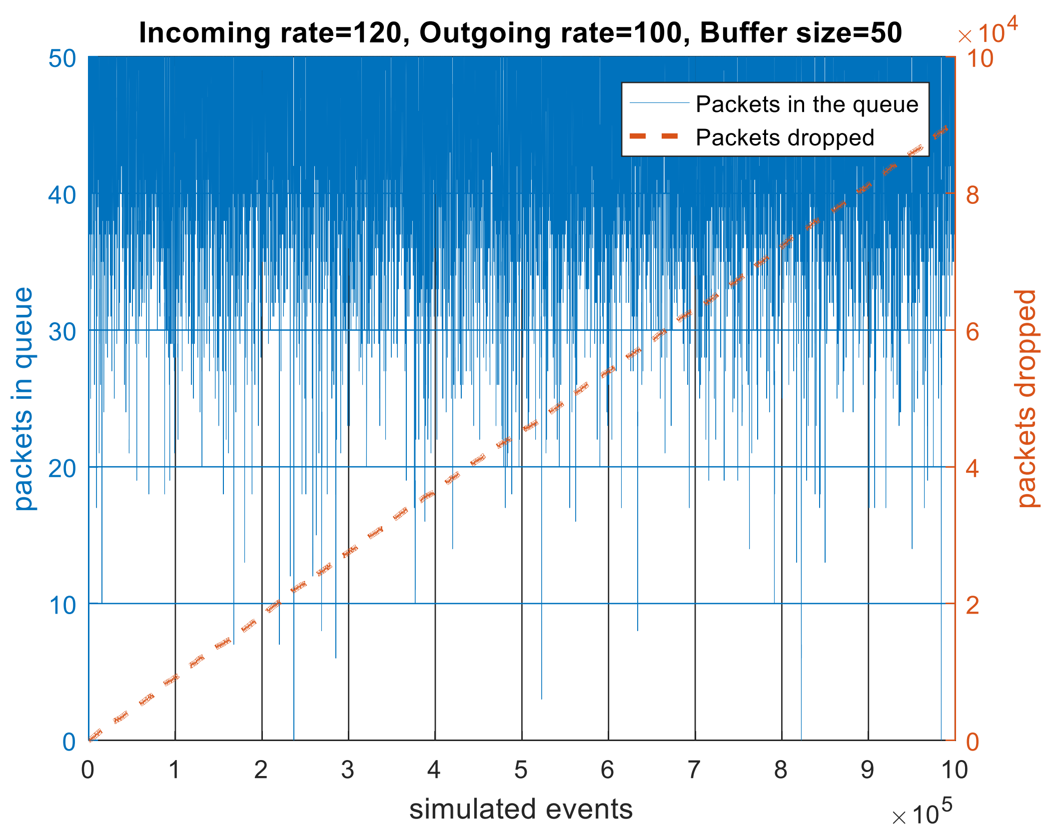


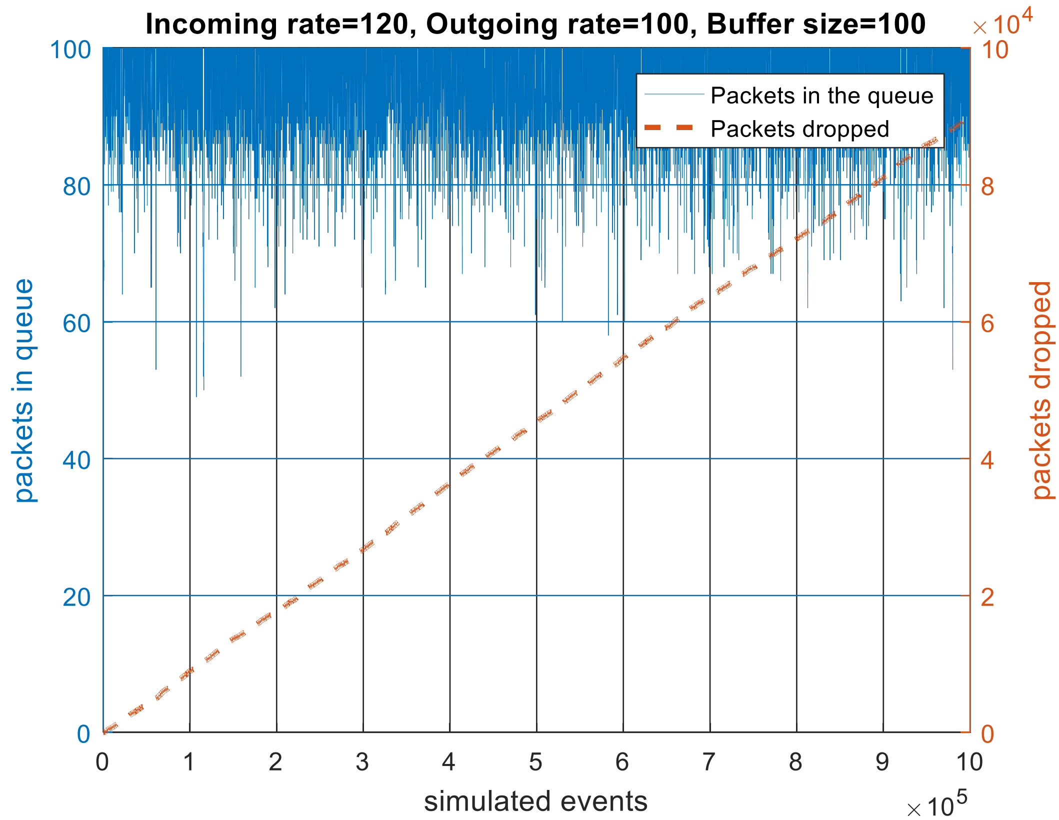
Incoming rate=120, Outgoing rate=50, Buffer size=50
packets in queue
packets dropped
simulated events
Packets in the queue
Packets dropped

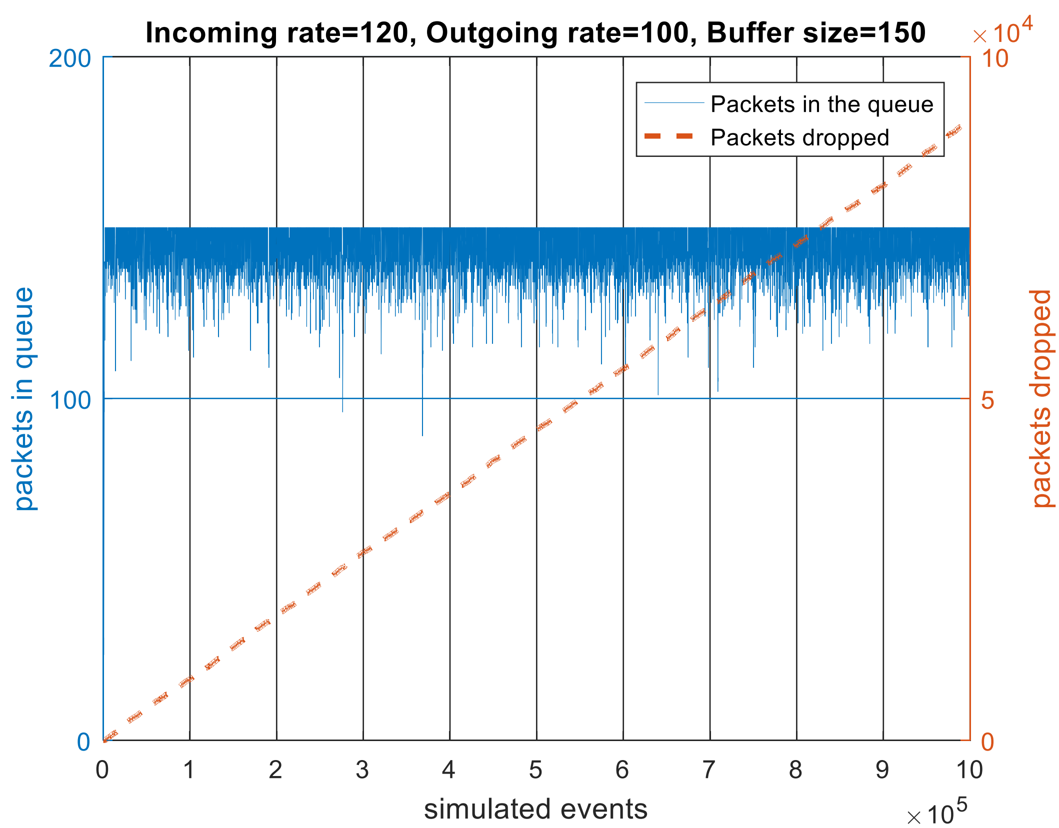


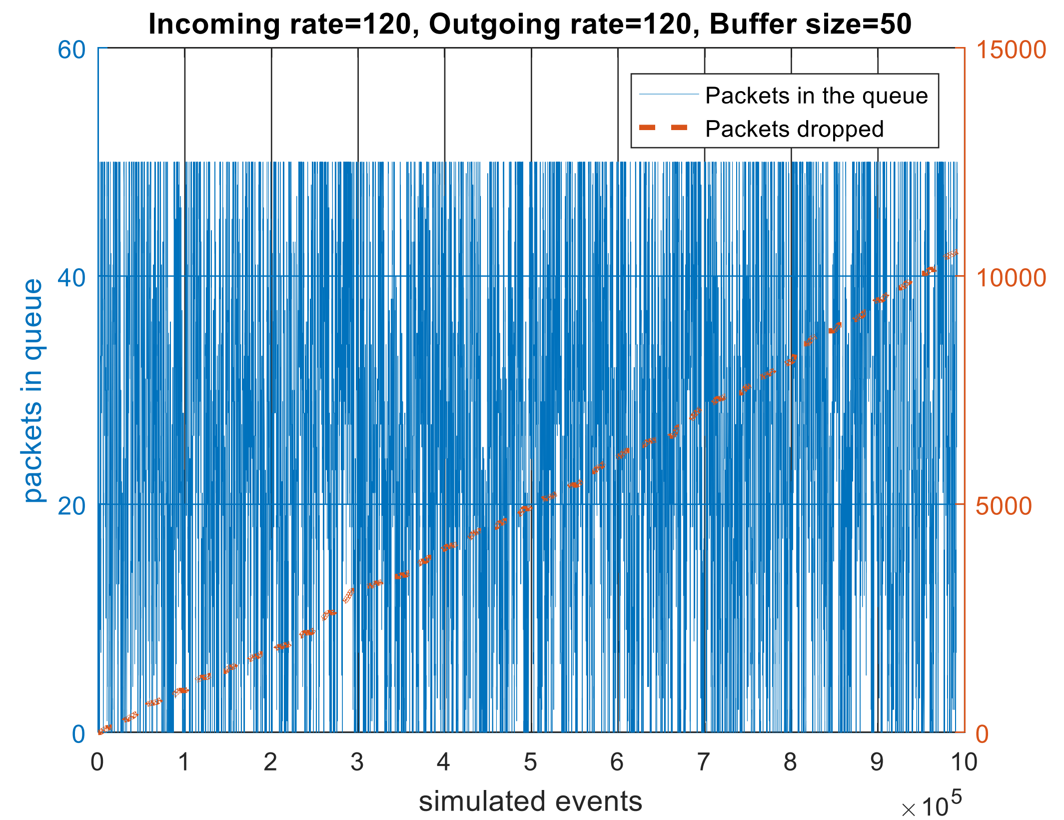
Incoming rate=120, Outgoing rate=50, Buffer size=100
packets in queue
packets dropped
simulated events
Packets in the queue
Packets dropped

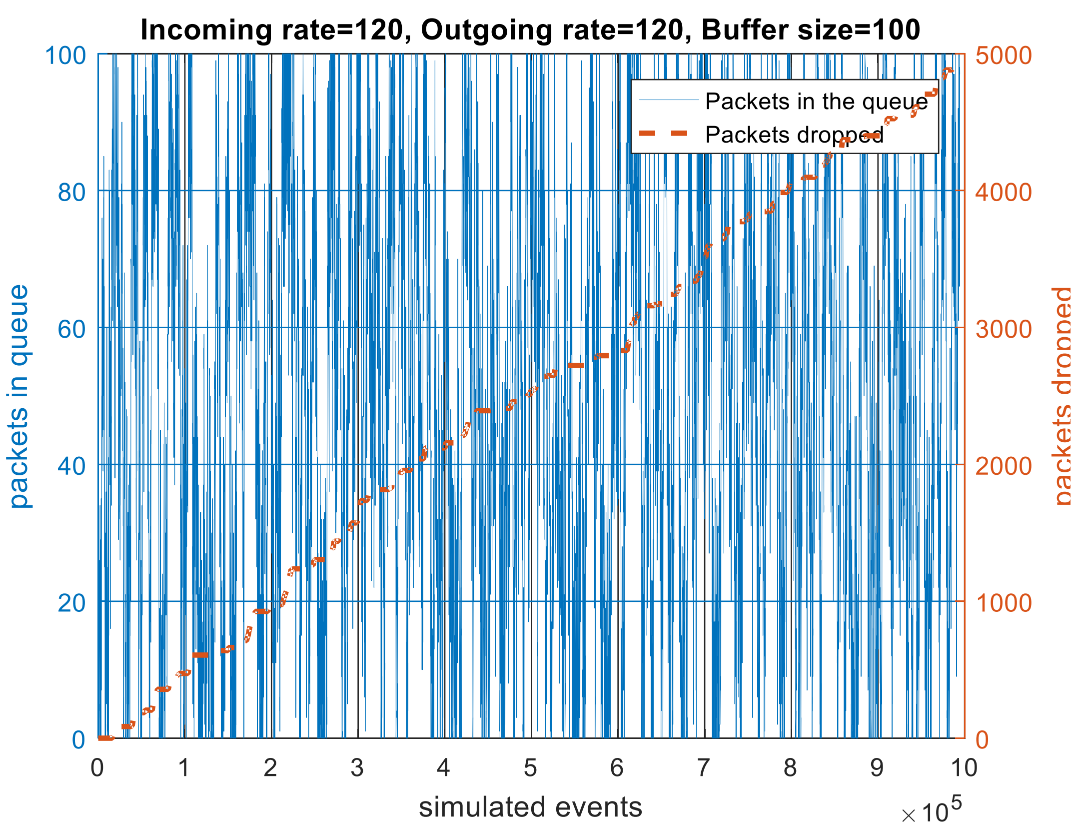

Incoming rate=120, Outgoing rate=50, Buffer size=150
packets in queue
packets dropped
simulated events
×10^5
×10^5
Packets in the queue
Packets dropped

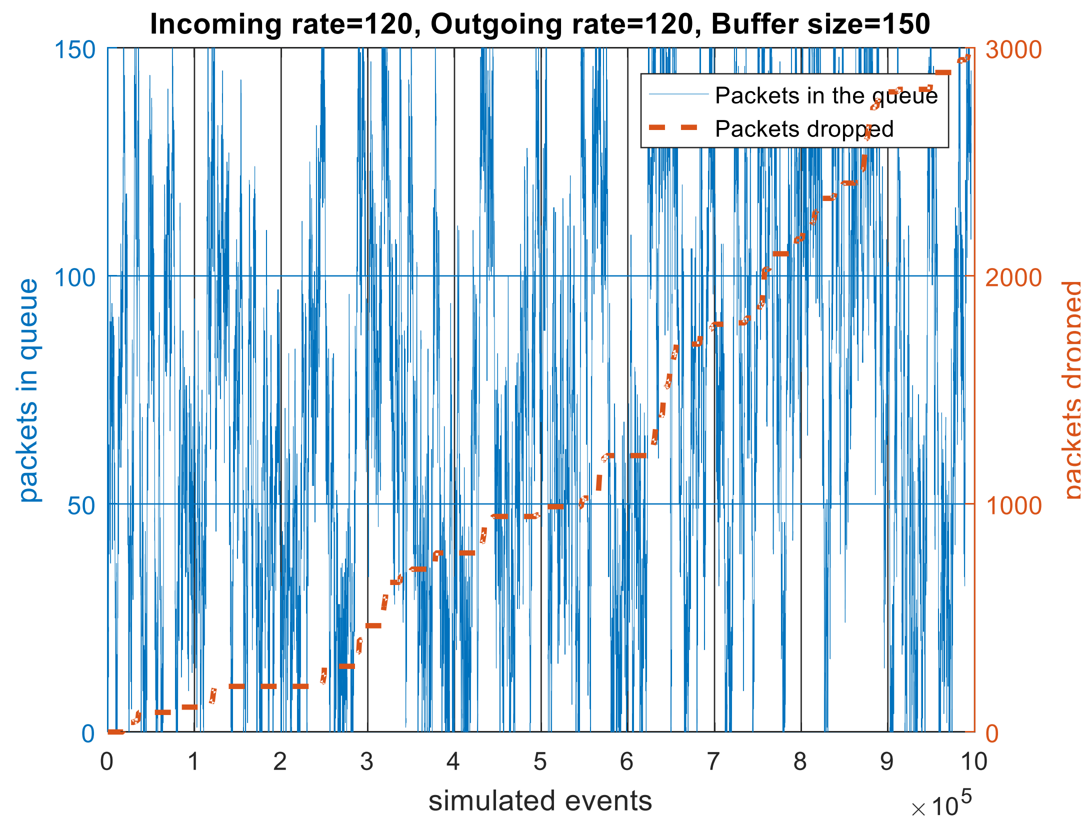

Incoming rate=120, Outgoing rate=100, Buffer size=50
packets in queue
packets dropped
simulated events
×10^4
×10^5
Packets in the queue
Packets dropped


Incoming rate=120, Outgoing rate=100, Buffer size=100
packets in queue
packets dropped
simulated events
Packets in the queue
Packets dropped
$\times 10^4$
$\times 10^5$
0
20
40
60
80
100
0
2
4
6
8
10
0
1
2
3
4
5
6
7
8
9
10


Incoming rate=120, Outgoing rate=100, Buffer size=150
packets in queue
packets dropped
simulated events
Packets in the queue
Packets dropped
$\times 10^4$
$\times 10^5$
0
100
200
0
5
10
0
1
2
3
4
5
6
7
8
9
10


Incoming rate=120, Outgoing rate=120, Buffer size=50
Packets in the queue
Packets dropped
packets in queue
simulated events
$\times 10^5$
0
20
40
60
0
5000
10000
15000
0
1
2
3
4
5
6
7
8
9
10


Incoming rate=120, Outgoing rate=120, Buffer size=100
Packets in the queue
Packets dropped
packets in queue
packets dropped
simulated events
$\times 10^5$
0
20
40
60
80
100
0
1000
2000
3000
4000
5000
0
1
2
3
4
5
6
7
8
9
10

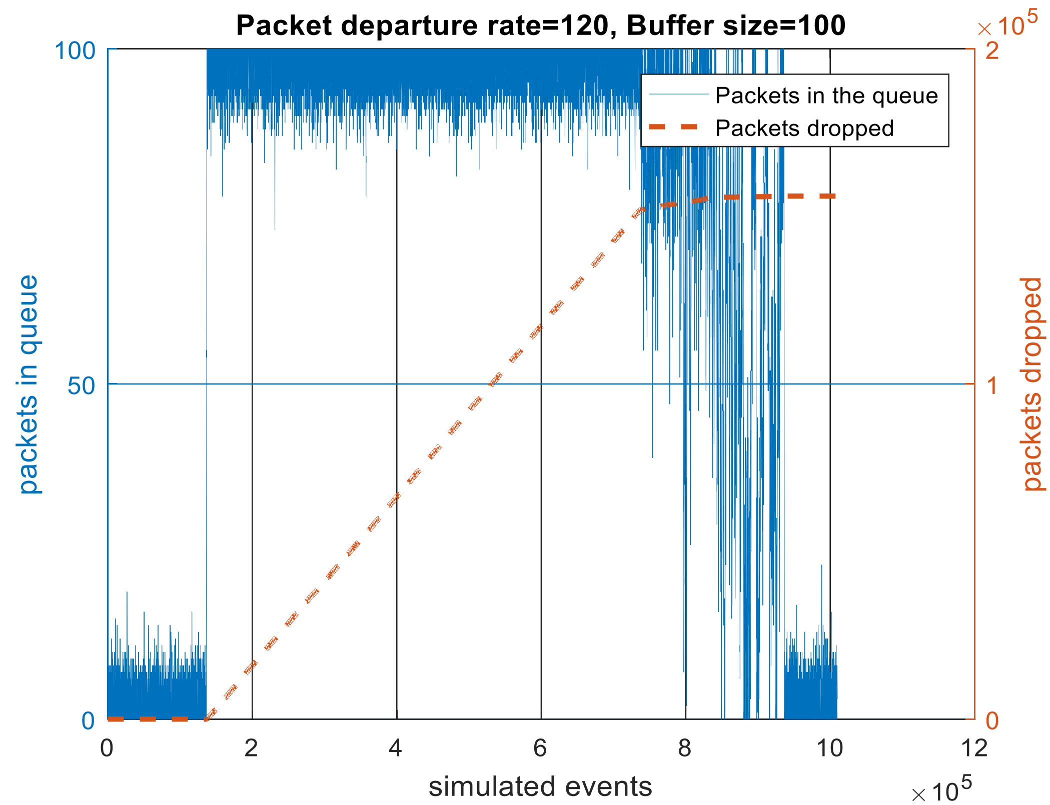


CASE2: Variable input rate

The departure rate μ = 120pkt/sec and buffer size n= 100 packets are set. The value of λ is varied here and the event is simulated for 1,000,000 times.

Events(%)	**λ (packets/second)**
0 – 10	70
10 – 70	200
70 – 80	130
80 – 90	120
90 – 100	70

<u>OUTPUT:</u>



CONCLUSION:

Thus the constant rate and the variable input rate are implemented and their plots are obtained. Using this kind of comparison shows that how the packets dropped and packets in queue vary with the input parameters.

APPENDIX A

CODE FOR CONSTANT RATES:

```
clc;
lambda = input('Enter the incoming rates: ');
mu = input('Enter the outgoing rates: ');
nbo = input('Enter the buffer sizes: ');
r = 1;
for m1 = 1:numel(lambda)
for m2 = 1:numel(mu)
for m3 = 1:numel(nbo)
out(r, 1:3) = [lambda(m1) mu(m2) nbo(m3)]; % possible combinations are done
r = r+1;
end
end
end
p_v=[]; %the number of packets in queue will be stored here
p_d=[]; %the number of dropped packets will be stored here
packet_in_queue=0; %initially packet in queue is considered as zero
packet_dropped=0; %initially packet dropped is considered as zero
for t=1:length(out)
lam=out(t,1);
mu1=out(t,2);
nb=out(t,3);
for event=1:1000000 % 1,000,000 events are simulated
x = rand;
if x <= lam/(lam+mu1)
if packet_in_queue < nb
packet_in_queue = packet_in_queue + 1; %packet is stored in queue when buffer
isn't full
p_v(end+1) = packet_in_queue; % packet in queue is modified
p_d(end+1) = pkt_dropped; % packet dropped is modified
else
pkt_dropped = pkt_dropped + 1; %packet is dropped when buffer is full
p_d(end+1) = pkt_dropped;% packet dropped is modified
p_v(end+1) = packet_in_queue;% packet in queue is modified
end
else
if packet_in_queue > 0
packet_in_queue = packet_in_queue - 1;
p_v(end+1) = packet_in_queue;
p_d(end+1) = pkt_dropped;
end
end
end
z=length(p_v);
m=1:1:z;
[ax,b1,b2] = plotyy(m,p_v,m,p_d);
b1.LineStyle = '-';
b2.LineStyle = '--';
b1.LineWidth = 0.2;
b2.LineWidth = 2;
title(['Incoming rate=',num2str(lam),', Outgoing rate=',num2str(mu1),',
Buffer size=',num2str(nb)]);
xlabel(ax(1),'simulated events');
```

```
ylabel(ax(1),'packets in queue');
ylabel(ax(2),'packets dropped');
grid on;
legend('Packets in the queue','Packets dropped');
pause(20);
pkt_dropped = 0;
packet_in_queue = 0;
p_d = [];
p_v = [];
end
```

APPENDIX B

CODE FOR VARIABLE INPUT RATE:

```
clc;
mu = 120;
n = 100;
p_v=[]; %the number of packets in queue will be stored here
p_d=[]; %the number of dropped packets will be stored here
packet_in_queue=0; %initially packet in queue is considered as zero
packet_dropped=0; %initially packet dropped is considered as zero
for t=1:1000000 %1,000,000 times its simulated
    percent=(t/1000000)*100;
    if percent<10
lambda=70;
x = rand;
if x <= lambda/(lambda+mu)
if packet_in_queue < n
packet_in_queue = packet_in_queue + 1; %packet is stored in queue when buffer
isn't full
p_v(end+1) = packet_in_queue; % packet in queue is modified
p_d(end+1) = packet_dropped; % packet dropped is modified
else
packet_dropped = packet_dropped + 1; %packet is dropped when buffer is full
p_d(end+1) = packet_dropped; % packet dropped is modified
p_v(end+1) = packet_in_queue; % packet in queue is modified
end
else
if packet_in_queue>0
packet_in_queue = packet_in_queue - 1;
end
end
p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
    else
if (10<=percent)&(percent<70)
lambda=200;
x = rand;
if x <= lambda/(lambda+mu)
if packet_in_queue < n
packet_in_queue = packet_in_queue + 1;
p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
else
packet_dropped = packet_dropped + 1; p_d(end+1) = packet_dropped; p_v(end+1)
= packet_in_queue;
end
else
if packet_in_queue>0
packet_in_queue = packet_in_queue - 1;
p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
end
end
elseif (70<=percent)&(percent<80)
lambda=130;
```

```
x = rand;
if x <= lambda/(lambda+mu)
if packet_in_queue < n
packet_in_queue = packet_in_queue + 1; p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
else
packet_dropped = packet_dropped + 1;
p_d(end+1) = packet_dropped;
p_v(end+1) = packet_in_queue;
end
else
if packet_in_queue>0
packet_in_queue = packet_in_queue - 1; p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
end
end
elseif (80<=percent)&(percent<90)
lambda=120;
x = rand;
if x <= lambda/(lambda+mu)
if packet_in_queue < n
packet_in_queue = packet_in_queue + 1; p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
else
packet_dropped = packet_dropped + 1;
p_d(end+1) = packet_dropped;
p_v(end+1) = packet_in_queue;
end
else
if packet_in_queue>0
packet_in_queue = packet_in_queue - 1; p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
end
end
else
lambda=70;
x = rand;
if x <= lambda/(lambda+mu)
if packet_in_queue < n
packet_in_queue = packet_in_queue + 1;
p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
else
packet_dropped = packet_dropped + 1; p_d(end+1) = packet_dropped; p_v(end+1)
= packet_in_queue;
end
else
if packet_in_queue>0
packet_in_queue = packet_in_queue - 1;
p_v(end+1) = packet_in_queue;
p_d(end+1) = packet_dropped;
end
end
end
end
end
figure;
```

```
z=length(p_v);
m=1:1:z;
[ax,b1,b2] = plotyy(m,p_v,m,p_d);
b1.LineStyle = '-';
b2.LineStyle = '--';
b1.LineWidth = 0.2;
b2.LineWidth = 2;
title(['Packet departure rate=',num2str(mu),', Buffer size=',num2str(n)]);
xlabel(ax(1),'simulated events');
ylabel(ax(1),'packets in queue');
ylabel(ax(2),'packets dropped');
grid on;
legend('Packets in the queue','Packets dropped');
```