ECE 642 – Fall 2006
Typical solution for Project #4

I Project description: Simulating a Queueing System with Feedback

In this project we simulate an M/M/1 queue with feedback at the output, such that a customer completing service either leaves the system, or (with fixed probability p) re-enters the system. See Bertsekas and Gallager, p. 228 for examples and analysis. For numerical results assume p=0.2 and 0.5

- Part a: Assume that successive service times for a customer that re-enters the queue are independent. Plot the mean queue and mean time in the system for an infinite-length queue. Compare simulation results with the expected analytical values.
- Part b: Now assume that the service times of those customers who re-enter the queue are not varied. Obtain the same plots as in Part a and compare the results.

II Analytical results & Simulation results

In this section, we will show the simulation results in graphs. The results include two part: the mean queue size vs. utilization and the mean delay vs. utilization. In each part, two cases will be plotted:one is for feedback probability $p = 0.2$ and the other is for feedback probability $p = 0.5$. Both independent service time and non-varied service time are considered for each case. For analytical values and considering λ' and ρ' as the aggregated arrival rate after feed back and the corresponding utilization:

$$\rho' = \frac{\rho}{1-p}$$

$$E[N] = \frac{\rho'}{1-\rho'} = \frac{\rho}{1-p-\rho} \Longrightarrow \rho < 1-p$$

- The queue size for $p = 0.5$ is plotted with $\rho = \{0.1, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45\}$. similarly, for $p = 0.2$ the queue size is plotted for $\rho = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.75\}$.

  The analytical and simulated values obtained are quite close. The analytical values are plotted with a dash line, the simulated values for independent service time are plotted with 'o' and the simulated values for non-varied service time are plotted with '*'.

- The Mean delay is obtained by using the Little's Formula, $E[T] = \frac{E[N]}{\lambda}$. The analytical result is obtained by using the above formula and substituting E[N] as calculated from analytical result. The analytical and simulated values obtained are quite close. The analytical values are plotted with a dash line, the simulated values for independent service time are plotted with 'o' and the simulated values for non-varied service time are plotted with '*'.

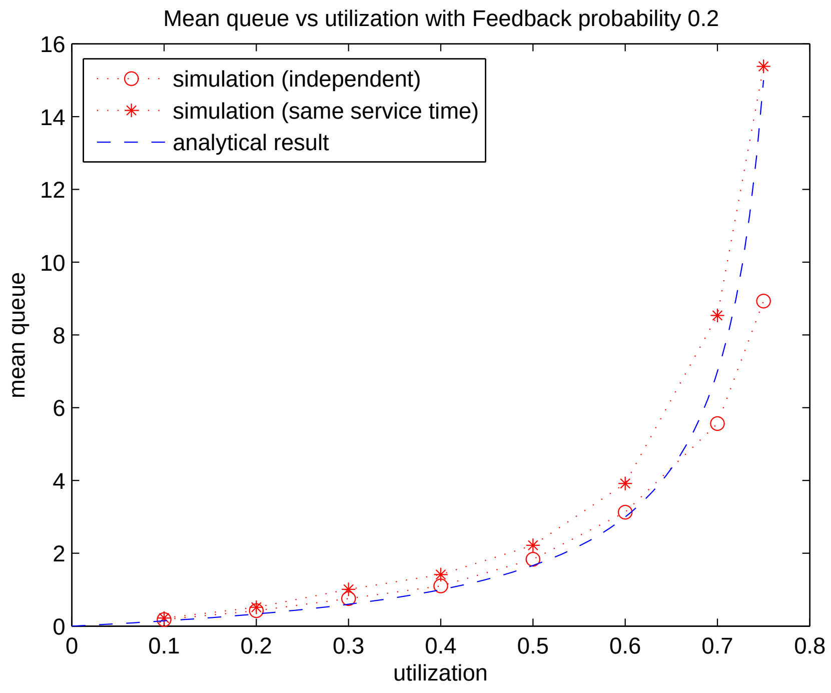

Figure 1: Mean queue and network utilization with feedback probability 0.2 for a Feedback Queueing system

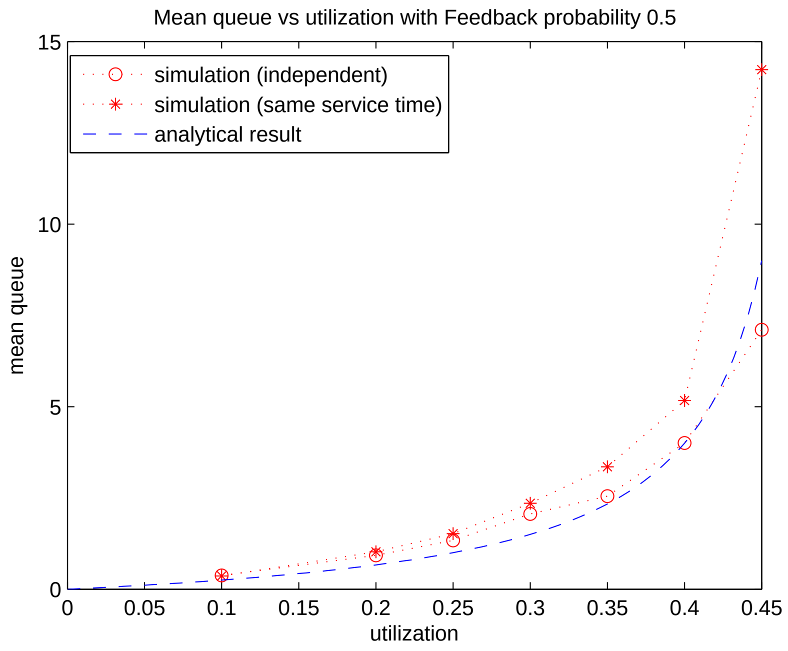

Figure 2: Mean queue and network utilization with feedback probability 0.5 for a Feedback Queueing system

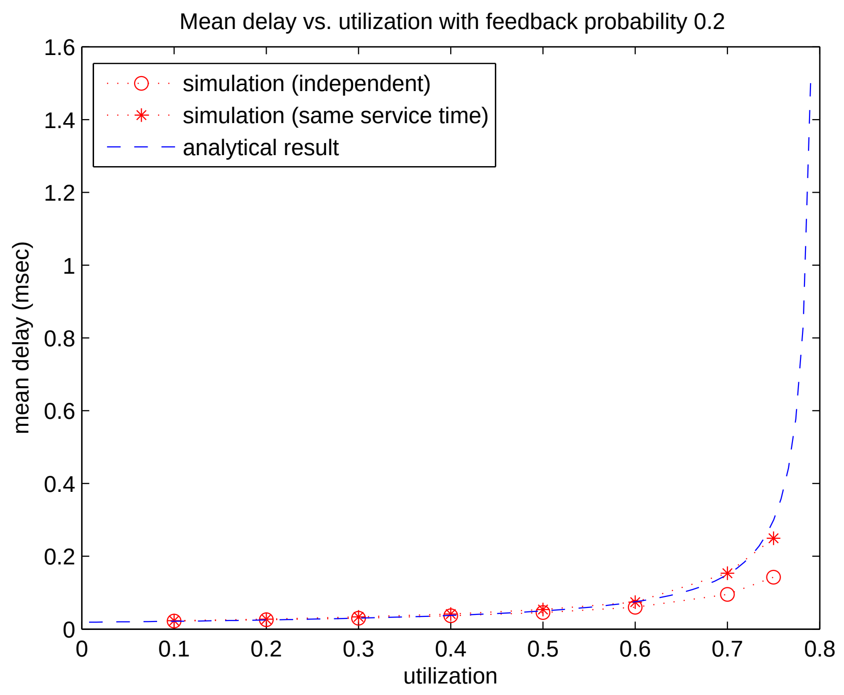

Figure 3: Mean delay and network utilization with feedback probability 0.2 for a Feedback Queueing system

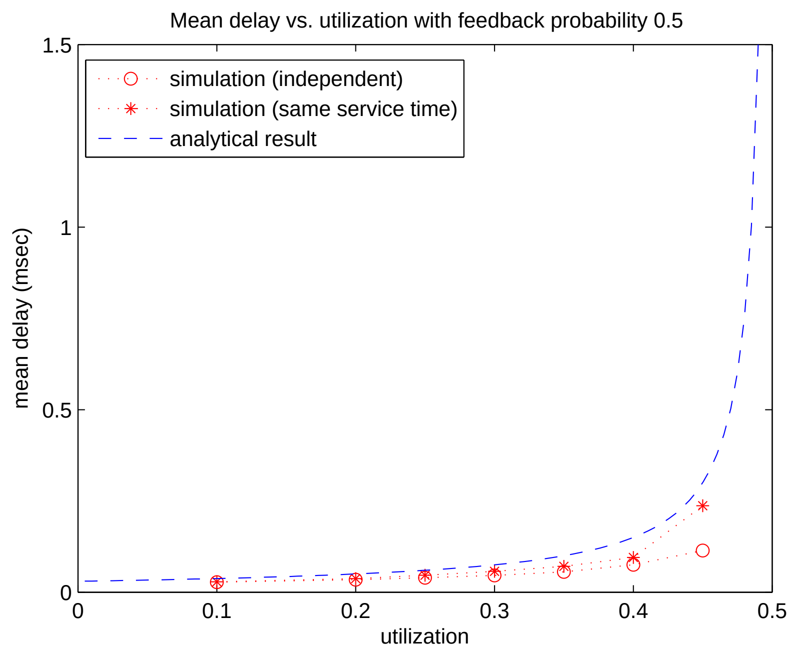

Figure 4: Mean delay and network utilization with feedback probability 0.5 for a Feedback Queueing system

III Conclusion

From the simulation, there is not much difference between the results of independent service time and non-varied service time for re-entered packets. But as ρ increases, the difference will increase especially when ρ approaches $1 - p$. And in this sensitive area where ρ is between $1 - p - 0.1$ and $1 - p$, the 'independent service time' case is closer to the analytical result of mean queue while the the 'non-varied service time' case is closer to analytical result of mean delay.

IV Program list

```
function [E_delay,E_N] =
delaythroughput(rho,seed1,seed2,p,style,NumberOfSamples);

% get the average service time and interarr time
E_service = 2325/(155*10^3); E_interarr = E_service/rho;

%generate two vector of random numbers with uniformly distribution
rand('state',seed1); u1 = rand(NumberOfSamples,1);
rand('state',seed2); u2 = rand(NumberOfSamples,1);

%obtain inter-arrival time random numbers and service time random numbers
Service_time = -E_service*log(u1); Interarr_time =
-E_interarr*log(u2);

%initialization: get the first depart time and all the arrival time
% vector to record arrival time for each packet (not including re-entered packets)
arrival = cumsum(Interarr_time);
% vector to record departure time for each packet(including re-entered packets)
depart(1) = arrival(1)+Service_time(1);

%will be the total number of packets pass the system including re-enter
new_NumberOfSamples = NumberOfSamples;
% new vector for arrival time including re-entered packets
new_arrival(1) = arrival(1);
% new vector for service time including re-entered packets
new_servicetime(1) = Service_time(1);
wait(1)=0; % waiting time in buffer for each packet (including those re-entered)
new_iterator = 2; % iterator for vector new_arrival and new_serivcetime
old_iterator = 2; % iterator for old arrival and service_time vector
depart_iterator = 1; % iterator for depart time vector

 % this loop will check for every departing packet
while (depart_iterator < new_NumberOfSamples)
    if( rand(1) <= p) % the packet will re-enter the queue
        % add all the packets that arrival before this departure time
        % to the new arrival and service_time vector
        for i = old_iterator:NumberOfSamples,
            % packets arrive before this departure time
            if(arrival(i) < depart(depart_iterator))
                % add into the new arrival vector
                new_arrival(new_iterator) = arrival(i);
                % add into the new service time vector
                new_servicetime(new_iterator) = Service_time(i);
                new_iterator = new_iterator + 1; % increase the iterator
            else
                % break when find the first packet that arrival
                % after this departure
                break;
            end;
        end;

        % add the re-enter packet into the new_arrival,
        % new_servicetime vector and increase the total sample number
        new_arrival(new_iterator) = depart(depart_iterator);
        if(style == 1) % for the service time is independent
            new_servicetime(new_iterator) = -E_service*log(rand(1));
        else % for the service time is the same with last time
            new_servicetime(new_iterator) = new_servicetime(depart_iterator);
        end;
        new_iterator = new_iterator + 1;
        new_NumberOfSamples = new_NumberOfSamples + 1;
        old_iterator = i;
    else % the packet will exit the system
        if(old_iterator < NumberOfSamples & length(new_arrival) <= length(depart))
            new_arrival(new_iterator) = arrival(old_iterator+1);
            new_servicetime(new_iterator) = Service_time(old_iterator+1);
            new_iterator = new_iterator + 1;
            old_iterator = old_iterator + 1;
        end;
    end;

    %get the next departure time and corresponding waiting time in buffer
    if(length(new_arrival) > length(depart))
        depart(depart_iterator+1) = max((depart(depart_iterator)),
            (new_arrival(depart_iterator+1))) + new_servicetime(depart_iterator+1);
        wait(depart_iterator+1) = max(0,
                    (depart(depart_iterator)-new_arrival(depart_iterator+1)));
        depart_iterator = depart_iterator + 1;
    end;
end;

% get the system delay for each packets including re_entered ones
for i=1:new_NumberOfSamples,
    delay(i) = wait(i) + new_servicetime(i);
end; E_delay = mean(delay)
```

```
%Calculate the mean number of packets in system, just copy from project2
%N_i = number of customers in the system at end of ith msec
%    = number of customers in system at (i-1)th msec (rest)
%       + number of customers arrive during ith msec (arrival)
%       - number of customers depart during ith msec (depart)
rest = 0; arrival = 0; dep = 0; low_bound1 = 1; low_bound2 = 1;
total_delay = 0; for i=1:max(depart)
    for j=low_bound1:new_NumberOfSamples,
        if(new_arrival(j)<=i)
            arrival = arrival + 1;
            total_delay = total_delay + delay(j);
        else
            low_bound1 = j;
            break;
        end;
    end;
    for k=low_bound2:new_NumberOfSamples,
        if(depart(k)<=i)
            dep = dep + 1;
        else
            low_bound2 = k;
            break;
        end;
    end;
    N(i) = rest + arrival - dep;
    E_T(i) = total_delay/low_bound1; %the mean delay over time
    rest = N(i);
    arrival = 0;
    dep = 0;
end;

E_N=mean(N)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%                              project 4                          %%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Description: In this project we simulate an M/M/1 queue with feedback at the output,    %%
%such that a customer completing service either leaves the system, or (with fixed pro      %
%-bability p) re-enters the sytem. See Bertsekas & Gallager, p.228 for examples and        %
%analysis. For numerical results assume p=0.2 and 0.5                                      %
%%Requirement: a, Assume that successive service times for a customer that re-enters the   %
%the queue are independent. Plot the mean queue and mean time in the system for an infinite%
%length queue. Compare simulation results with the expected analytical values. b,Now assume%
%that the service times of those customers who re-enter the queue are not varied.Obtain the%
```

```
%same plots in Part a and compare the results.                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NumberOfSamples = 5000;

%for p=0.2
rho1=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.75];

%for p=0.5
rho2=[0.1,0.2,0.25,0.3,0.35,0.4,0.45];

% calculate the average delay and mean queue for p=0.2
for i=1:8,
    [E_delay1(i),E_N1(i)] = delnumthroughput(rho1(i),9,10,0.2,1,NumberOfSamples);
    [E_delay11(i),E_N11(i)] = delnumthroughput(rho1(i),11,12,0.2,1,NumberOfSamples);
    [E_delay12(i),E_N12(i)] = delnumthroughput(rho1(i),21,22,0.2,1,NumberOfSamples);
    [E_delay13(i),E_N13(i)] = delnumthroughput(rho1(i),36,37,0.2,1,NumberOfSamples);
    [E_delay14(i),E_N14(i)] = delnumthroughput(rho1(i),78,79,0.2,1,NumberOfSamples);

    [E_delay2(i),E_N2(i)] = delnumthroughput(rho1(i),9,10,0.2,2,NumberOfSamples);
    [E_delay22(i),E_N22(i)] = delnumthroughput(rho1(i),11,12,0.2,2,NumberOfSamples);
    [E_delay23(i),E_N23(i)] = delnumthroughput(rho1(i),21,22,0.2,2,NumberOfSamples);
    [E_delay24(i),E_N24(i)] = delnumthroughput(rho1(i),36,37,0.2,2,NumberOfSamples);
    [E_delay25(i),E_N25(i)] = delnumthroughput(rho1(i),78,79,0.2,2,NumberOfSamples);
end;

E_delay1 = (E_delay1+E_delay11+E_delay12+E_delay13+E_delay14)/5;
E_N1 = (E_N1+E_N11+E_N12+E_N13+E_N14)/5; E_delay2 =
(E_delay2+E_delay22+E_delay23+E_delay24+E_delay25)/5; E_N2 =
(E_N2+E_N22+E_N23+E_N24+E_N25)/5


% calculate the average delay and mean queue for p=0.5
for i=1:7,
    [E_delay3(i),E_N3(i)] = delnumthroughput(rho2(i),9,10,0.5,1,NumberOfSamples);
    [E_delay31(i),E_N31(i)] = delnumthroughput(rho2(i),11,12,0.5,1,NumberOfSamples);
    [E_delay32(i),E_N32(i)] = delnumthroughput(rho2(i),21,22,0.5,1,NumberOfSamples);
    [E_delay33(i),E_N33(i)] = delnumthroughput(rho2(i),36,37,0.5,1,NumberOfSamples);
    [E_delay34(i),E_N34(i)] = delnumthroughput(rho2(i),78,79,0.5,1,NumberOfSamples);

    [E_delay4(i),E_N4(i)] = delnumthroughput(rho2(i),9,10,0.5,2,NumberOfSamples);
    [E_delay41(i),E_N41(i)] = delnumthroughput(rho2(i),11,12,0.5,2,NumberOfSamples);
    [E_delay42(i),E_N42(i)] = delnumthroughput(rho2(i),21,22,0.5,2,NumberOfSamples);
    [E_delay43(i),E_N43(i)] = delnumthroughput(rho2(i),36,37,0.5,2,NumberOfSamples);
    [E_delay44(i),E_N44(i)] = delnumthroughput(rho2(i),78,79,0.5,2,NumberOfSamples);

end; E_delay3 =
(E_delay3+E_delay31+E_delay32+E_delay33+E_delay34)/5; E_N3 =
(E_N3+E_N31+E_N32+E_N33+E_N34)/5; E_delay4 =
(E_delay4+E_delay41+E_delay42+E_delay43+E_delay44)/5; E_N4 =
(E_N4+E_N41+E_N42+E_N43+E_N44)/5;

%Plot Mean Size and compare to analytical result
% Plot simulated mean number of packets in system for p=0.2
   figure(1);
   plot(rho1,E_N1,'ro:',rho1,E_N2,'r*:');
   grid on;

% Plot predicted value for comparision fir p=0.2
   hold on;
   ro=linspace(0,.75);
   r1=ro/(1-0.2);
   ana=r1./(1-r1);
   plot(ro,ana,'--')
   xlabel('utilization');
   ylabel('mean queue');
   title('Mean queue vs utilization with Feedback probability 0.2');
   legend('simulation (independent)','simulation (same service time)','analytical result');
   grid;

% Plot simulated mean number of packets in system for p=0.5
   figure(2);
   plot(rho2,E_N3,'ro:',rho2,E_N4,'r*:');
   grid on;

% Plot predicted value for comparision fir p=0.2
   hold on;
   ro=linspace(0,.45);
   r1=ro/(1-0.5);
   ana=r1./(1-r1);
   plot(ro,ana,'--')
   xlabel('utilization');
   ylabel('mean queue');
   title('Mean queue vs utilization with Feedback probability 0.5');
   legend('simulation (independent)','simulation (same service time)','analytical result');
   grid;


  mu=155000/2325; %time unit is msec
% Plot Mean delay and compare to analytical results
% Plot simulated average delay for p=0.2
   figure(3);
   plot(rho1,E_delay1,'ro:',rho1,E_delay2,'r*:');
   grid on;
```

```
% Plot predicted value for comparision for p=0.2
   hold on;
   ro=linspace(0,.79);
   r1=ro/(1-0.2);
   for j=1:100
      D(j)=(r1(j)/(1-r1(j)))/(ro(j)*mu);
   end;
   plot(ro,D,'--')
   xlabel('utilization');
   ylabel('mean delay');
   title('Mean delay vs. utilization with feedback probability 0.2');
   legend('simulation (independent)','simulation (same service time)','analytical result');
   grid;

% Plot simulated average delay for p=0.5
   figure(4);
   plot(rho2,E_delay3,'ro:',rho2,E_delay4,'r*:');
   grid on;

% Plot predicted value for comparision for p=0.5
   hold on;
   ro=linspace(0,.49);
   r1=ro/(1-0.5);
   for j=1:100
      D(j)=(r1(j)/(1-r1(j)))/(ro(j)*mu);
   end;
   plot(ro,D,'--')
   xlabel('utilization');
   ylabel('mean delay');
   title('Mean delay vs. utilization with feedback probability 0.5');
   legend('simulation (independent)','simulation (same service time)','analytical result');
   grid;
```