Weining Bao weiningb@usc.edu 8287723653

Project 9: CTMC and Queues

A computer lab has M = 120 computers. From time to time a computer stops working and requires intervention by a technician to resolve the problem. For any individual computer, the time until it stops working (after it is initially switched on or after being repaired) is exponentially distributed with mean T = 4 days (i.e. 96 hours). There are K technicians available to work on the computers (for simplicity assume that a technician works 24 hours a day). We will consider 3 possible distributions for the time to repair a computer but in all cases the mean time to repair a computer is X = 1 hour.

Case A – the distribution of the repair time is exponentially distributed (this corresponds to the Machine Repairman model discussed in class).
Case B – the time to repair the computer is always exactly one hour.
Case C - with probability 0.8, the time to repair the computer is 10 minutes (a reboot) and with probability 0.2, the time to repair the computer is 4 hours and 20 minutes (a hardware problem).

Compare these 3 cases for one or two technicians (i.e. K = 1 or 2) to find the distribution (and mean) of the number of computers not working.

Matlab Simulation

with code at last

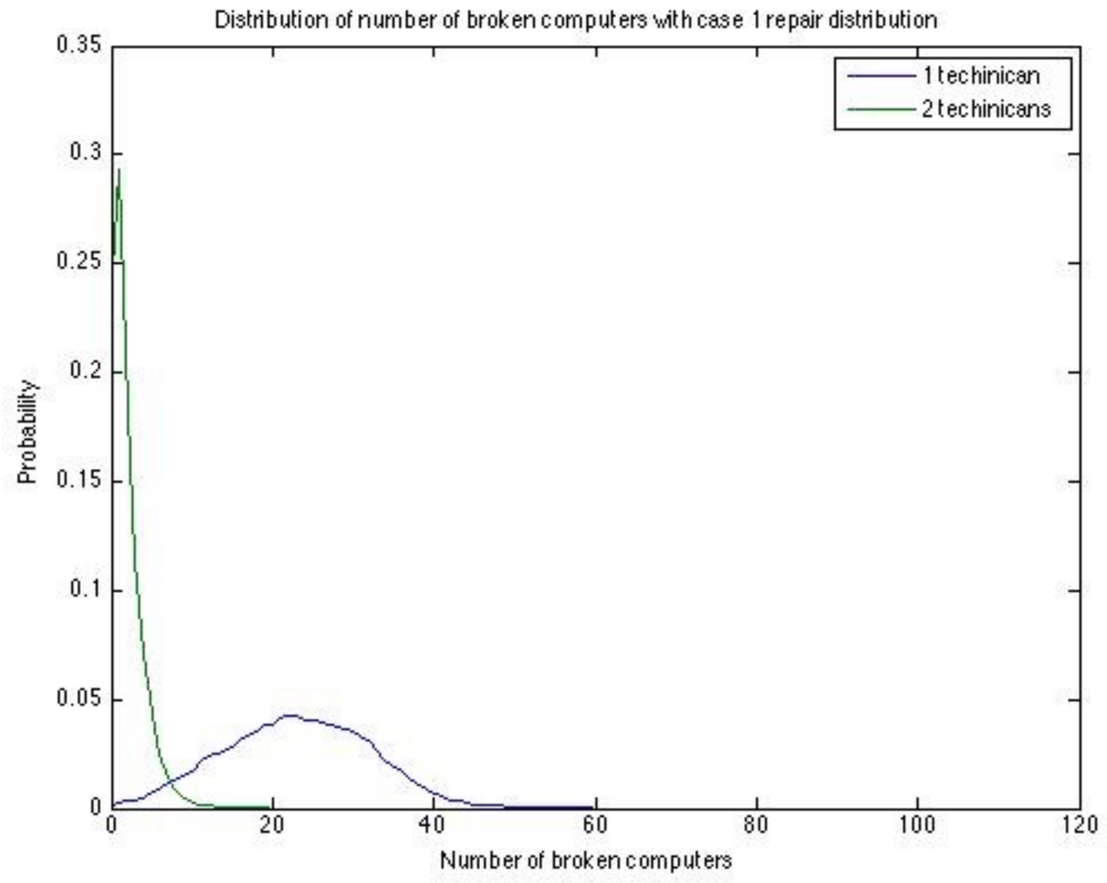
The simulation code follows the structure of the sample code at the end of the note 9 with event driven simulation. However, no numbers of arrivals or departures are recorded because they are both kind of events. Instead, one variable is recorded as event number where event can be one computer breaks or is repaired. The simulation ends till the number of events reaches one preset value, like 100,000 in this simulation. Larger value leads to closer result to theoretical distribution but also longer simulation time. Besides the variable to record the number of events happened, the simulation time passed is also recorded in a discrete way.

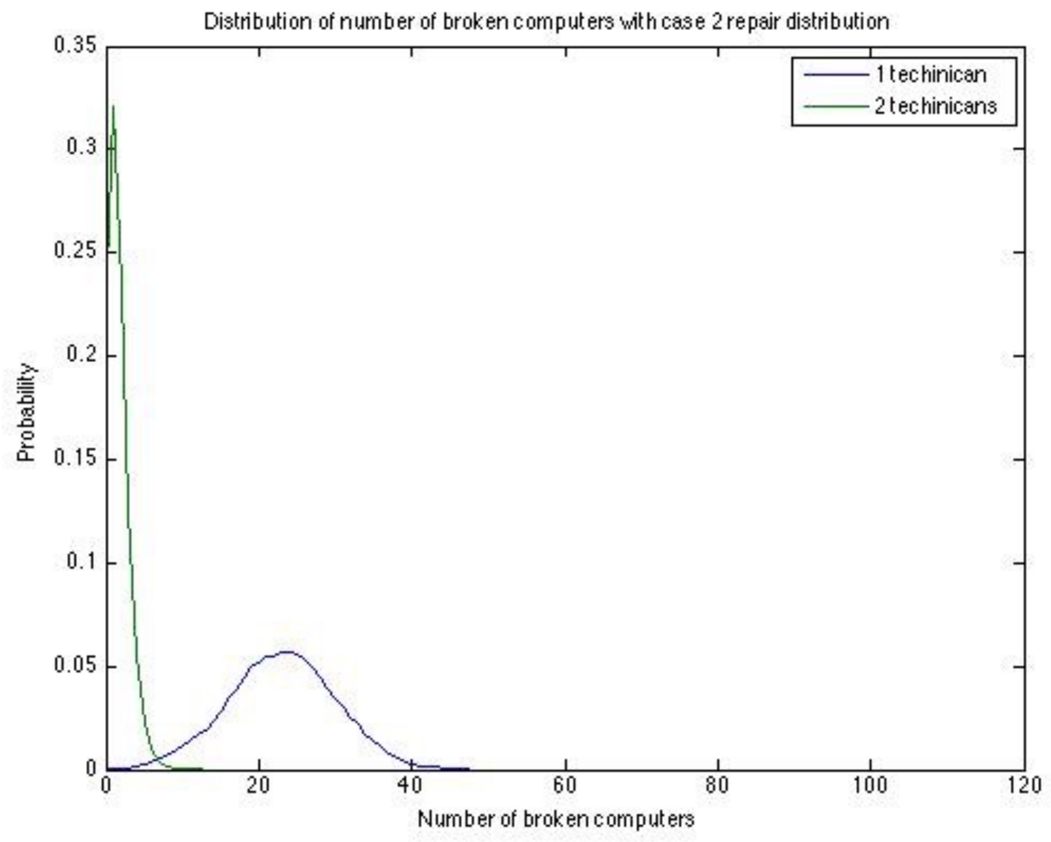
For one technician, only one repaired time is recorded. For two technicians, two repaired times are recorded and processed separately. One event always leads to the change of state, which is the number of broken computers. As long as one event happens, post state and the timestamp of the event is recorded in two vectors. Two vectors have the same dimension, that is preset number of events to be simulated. If the state vector is s, the time vector is t and t(k) means the kth element of t, the time spent in state s(i) is t(i+1) - t(i). Two vectors help to accumulate the time spent in different states. After normalized by the total time of simulation, the distributions are plotted. Here are the results of mean and plots of distribution:

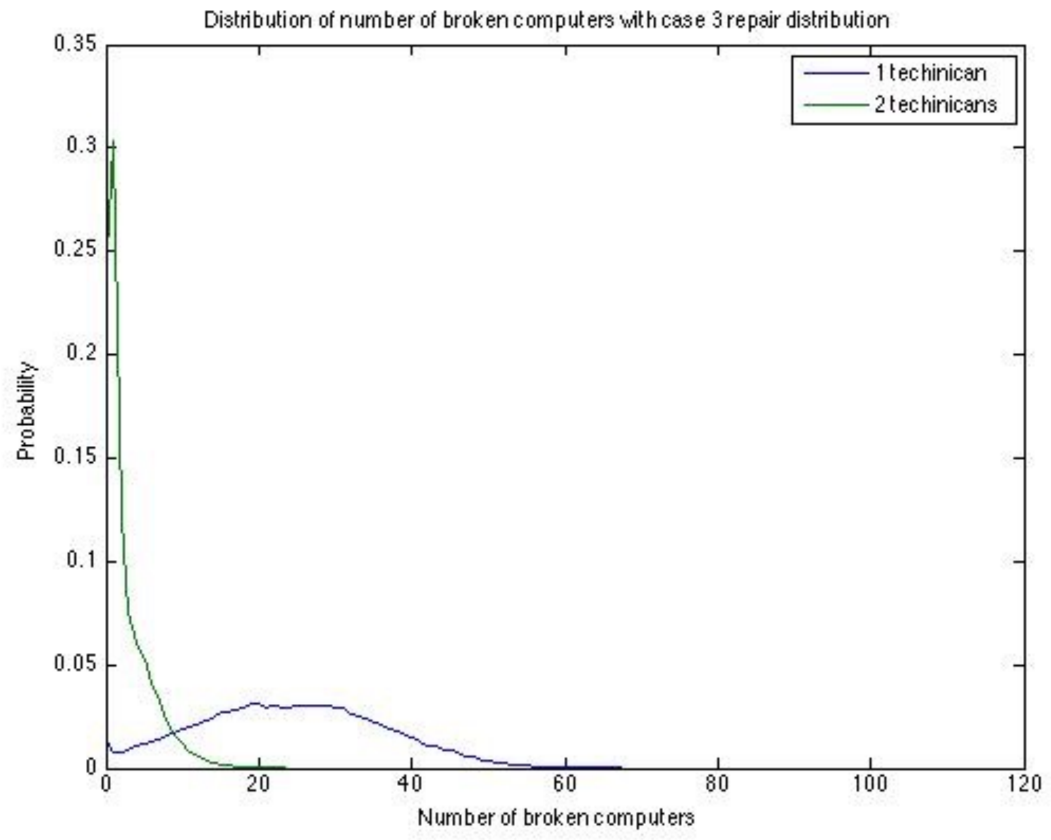
```
Which case for repair distribution? 1 for A, 2 for B, 3 for C: 1

 1 techinican:
The mean of the number of broken computers with case 1, 1 techinican is 23.037211.

 2 techinicans:
The mean of the number of broken computers with case 1, 2 techinican is 1.928923.
```

```
Which case for repair distribution? 1 for A, 2 for B, 3 for C: 2

 1 techinican:
The mean of the number of broken computers with case 2, 1 techinican is 23.010211.

 2 techinicans:
The mean of the number of broken computers with case 2, 2 techinican is 1.602124.

Which case for repair distribution? 1 for A, 2 for B, 3 for C: 3

 1 techinican:
The mean of the number of broken computers with case 3, 1 techinican is 24.093461.

 2 techinicans:
The mean of the number of broken computers with case 3, 2 techinican is 2.493069.
```

From the results it can be found that the mean of broken computers with 2 technicians is much smaller than the mean with only 1. This is reasonable because more technicians mean more possibility to repair. Three cases of repair distribution have same mean of 1 but different variances, which are 1, 0 and 5/3, respectively. This leads to the wider distribution and larger mean for case c > case a > case b. These are all demonstrated in the plots.




Distribution of number of broken computers with case 2 repair distribution
1 techinican
2 techinicans
Probability
Number of broken computers
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
20
40
60
80
100
120


Distribution of number of broken computers with case 3 repair distribution
1 techinican
2 techinicans
Probability
Number of broken computers
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
20
40
60
80
100
120

Code:

```
% Initialization
c=input('Which case for repair distribution? 1 for A, 2 for B, 3 for C: ');
no_coms=120; % number of computers
fprintf('\n 1 techinican: \n');
state=0; % no computer broken
sim_time=0; % time starts from 0
no_events=100000; % simulate 100000 events, broken or repair
states=zeros(1,no_events); % record of sequential state change
times=zeros(1,no_events); % timestamp of sequential state change
%
% generate initial arrival and departure
%
event_no=1;
lambda=1/96; % mean of 96 hours
mu=1; % mean of 1 hour
states(event_no)=state;
times(event_no)=sim_time;
next_break_time=-log(rand(1,1))/((no_coms-state)*lambda);
next_repair_time=next_break_time+1; % no repair at state 0
%
% Main simulation loop
%
while event_no<no_events
   % get next event
   event_no=event_no+1;
   if (next_break_time<next_repair_time)
      % arrival
      sim_time=next_break_time;
      if (state==0)
         if (c==1)
            next_repair_time=sim_time-log(rand(1,1))/mu;
         elseif (c==2)
            next_repair_time=sim_time+1;
         else
            if (rand()<0.8)
               next_repair_time=sim_time+1/6;
            else
               next_repair_time=sim_time+4+1/3;
            end
         end
      end
      state=state+1;
      states(event_no)=state;
      times(event_no)=sim_time;
      next_break_time=sim_time-log(rand(1,1))/((no_coms-state)*lambda);
```

```
    else
        %departure
        sim_time=next_repair_time;
        state=state-1;
        if (state>0)
            if (c==1)
                next_repair_time=sim_time-log(rand(1,1))/mu;
            elseif (c==2)
                next_repair_time=sim_time+1;
            else
                if (rand()<0.8)
                    next_repair_time=sim_time+1/6;
                else
                    next_repair_time=sim_time+4+1/3;
                end
            end
        else
            next_repair_time=next_break_time+1;
        end
        states(event_no)=state;
        times(event_no)=sim_time;
    end
end
%
% output
%
for i=no_events:-1:2 % generate the time in sequential states
    times(i)=times(i)-times(i-1);
end
dist=zeros(1,no_coms+1); % distribution of different state
for i=1:(event_no-1)
    dist(states(i)+1)=dist(states(i)+1)+times(i+1); % state i <-> dist(i+1)
end
dist=dist/sim_time;
figure;
plot(0:no_coms, dist); hold all;
title(['Distribution of number of broken computers with case ',num2str(c),...
      ' repair distribution']);
xlabel('Number of broken computers');
ylabel('Probability');
mean=0;
for i=1:no_coms+1
    mean=mean+(i-1)*dist(i);
end
fprintf('The mean of the number of broken computers with case %d, 1 techinican is %f.\n', c,
mean);

fprintf('\n 2 techinicans: \n');
state=0; % no computer broken
```

```
sim_time=0; % time starts from 0
states=zeros(1,no_events); % record of sequential state change
times=zeros(1,no_events); % timestamp of sequential state change
%
% generate initial arrival and departure
%
event_no=1;
lambda=1/96; % mean of 96 hours
mu=1; % mean of 1 hour
states(event_no)=state;
times(event_no)=sim_time;
next_break_time=-log(rand(1,1))/((no_coms-state)*lambda);
next_repair_time=zeros(1,2); % repaired time for 2 techinicans
next_repair_time(1)=next_break_time+1; % no repair at state 0
next_repair_time(2)=next_break_time+2; % no repair at state 0
%
% Main simulation loop
%
while event_no<no_events
    % get next event
    event_no=event_no+1;
    if (next_repair_time(1)<next_repair_time(2))
        m=1;
    else
        m=2;
    end
    if (next_break_time<next_repair_time(m))
        % arrival
        sim_time=next_break_time;
        if (state==0)
            if (c==1)
                next_repair_time(1)=sim_time-log(rand(1,1))/mu;
                next_repair_time(2)=next_repair_time(1)+1;
            elseif (c==2)
                next_repair_time(1)=sim_time+1;
                next_repair_time(2)=next_repair_time(1)+1;
            else
                if (rand()<0.8)
                    next_repair_time(1)=sim_time+1/6;
                else
                    next_repair_time(1)=sim_time+4+1/3;
                end
                next_repair_time(2)=next_repair_time(1)+1;
            end
        elseif (state==1)
            if (c==1)
                next_repair_time(2)=sim_time-log(rand(1,1))/mu;
            elseif (c==2)
                next_repair_time(2)=sim_time+1;
            else
                if (rand()<0.8)
                    next_repair_time(2)=sim_time+1/6;
                else
                    next_repair_time(2)=sim_time+4+1/3;
                end
            end
        end
        state=state+1;
        states(event_no)=state;
        times(event_no)=sim_time;
        next_break_time=sim_time-log(rand(1,1))/((no_coms-state)*lambda);
    else
        %departure
        sim_time=next_repair_time(m);
        state=state-1;
        if (state>1)
            if (c==1)
                next_repair_time(m)=sim_time-log(rand(1,1))/mu;
            elseif (c==2)
                next_repair_time(m)=sim_time+1;
            else
                if (rand()<0.8)
                    next_repair_time(m)=sim_time+1/6;
                else
                    next_repair_time(m)=sim_time+4+1/3;
                end
            end
        elseif (state==1)
            next_repair_time(m)=0; % move the only next_repair_time to index 1
            next_repair_time(1)=sum(next_repair_time);
            next_repair_time(2)=next_repair_time(1)+1;
        else
            next_repair_time(1)=next_break_time+1; % no repair at state 0
            next_repair_time(2)=next_break_time+2; % no repair at state 0
        end
        states(event_no)=state;
        times(event_no)=sim_time;
    end
end
%
% output
%
for i=no_events:-1:2 % generate the time in sequential states
    times(i)=times(i)-times(i-1);
end
dist=zeros(1,no_coms+1); % distribution of different state
for i=1:(event_no-1)
    dist(states(i)+1)=dist(states(i)+1)+times(i+1); % state i <-> dist(i+1)
```

```
end
dist=dist/sim_time;
plot(0:no_coms, dist); hold off;
legend('1 techinican', '2 techinicans');
mean=0;
for i=1:no_coms+1
   mean=mean+(i-1)*dist(i);
end
fprintf('The mean of the number of broken computers with case %d, 2 techinican is %f.\n', c,
mean);
```