Solution to Homework #1

1. (20 points.) Write a smpl simulation program for an M/M/3/10 queueing system to compute the (1) customer turned-away probability, (2) average number of customers waiting in the system (excluding the ones in service), (3) response time per client and (4) throughput, assuming that the arrival rate $\lambda = 5$ and the service rate $\mu = 2$ customers/second. Make sure that the reported response time is within 95% level of confidence with 5% confidence accuracy based on the batch mean analysis method. Also report the confidence accuracy obtained from your simulation program. Turn in the smpl program source code and output in a hardcopy.

An example output is:

```
batch 1 mean = 0.807
batch 2 mean = 0.905
batch 3 mean = 0.761
batch 4 mean = 0.807
batch 5 mean = 0.775
batch 6 mean = 0.895
batch 7 mean = 0.913
batch 8 mean = 0.775
batch 9 mean = 0.834
batch 10 mean = 0.907, rel. HW = 0.053
batch 11 mean = 0.947, rel. HW = 0.053
batch 12 mean = 0.773, rel. HW = 0.051
batch 13 mean = 0.828, rel. HW = 0.046

(1) Customer turned-away probability is 0.041092
(2) Average number of customers waiting is 1.641433
(3) Mean Response Time is 0.840508 with confidence accuracy 0.046470
(4) Throughput is 4.799136
```

This can be obtained from the following code:

```c
#include "smpl.h"
define TOKENS 1000
define TRUE 1
define FALSE 0
main()
{
    real Ta=0.2,Ts=0.5,mean,hw;
    int tk_id=0,customer=0,event,server,nb, n, rejected=0, completed =0;
    real ts[TOKENS]; /* start time stamp */
    real TotalTimeElapsed, rejectProb, queueLenght, X;
    int cont=TRUE;
```
int totalCustomersArrived = 0;
smpl(0,"M/M/3/10 Queue");
init_bm(200,2000); /* let m0 be 200 and mb be 2000 observations */
server=facility("server",3);
schedule(1,0.0,tk_id);
while (cont)
{
  cause(&event,&customer);
  switch(event)
  {
  case 1: /* arrival */
    totalCustomersArrived++;
    n = inq(server);
    if(n<7) /* there is still room to accept this client */
    {
      ts[customer] = time();
      schedule(2, 0.0, customer);
    }
    else { rejected++; }
    if (++tk_id >= TOKENS) tk_id=0;
    schedule(1, expntl(Ta), tk_id);
    break;
  case 2: /* request server */
    if (request(server,customer,0)==0) then
      schedule(3, expntl(Ts), customer);
    break;
  case 3: /* release server */
    release(server,customer);
    if (obs(time()-ts[customer]) == 1) cont = FALSE;
    break;
  }
}
TotalTimeElapsed = time();
rejectProb = rejected/(real)(totalCustomersArrived);
printf("(1)Customer turned-away probability is %f\n", rejectProb);
queueLenght =Lq(server);
printf("(2)Average number of customers waiting is %f\n", queueLenght);
civals(&mean, &hw, &nb);
printf("(3)Mean Response Time is %f with confidence accuracy %f\n", mean, hw/mean);
completed = totalCustomersArrived - rejected; /*Serviced customers*/
X = (real)(completed)/TotalTimeElapsed;
printf("(4)Throughput is %f\n", X);
2. (10 points.) Assume that four components obeying the exponential failure law have failure rates of \( \lambda_1, \lambda_2, \lambda_3 \) and \( \lambda_4 \), respectively; that is, \( R_1(t) = e^{-\lambda_1 t}, R_2(t) = e^{-\lambda_2 t}, R_3(t) = e^{-\lambda_3 t}, \) and \( R_4(t) = e^{-\lambda_4 t} \). Derive a mathematical expression for the system reliability of a 3-out-of-4 system using these four components based on the minimal cut set method. Note: you must use the minimal cut set method covered in the lecture.

**Answer:**

Let \( \phi(t), X_1(t), X_2(t), X_3(t), \) and \( X_4(t) \), be binary random variables representing the status of the system, component 1, component 2, component 3 and component 4, respectively, at time \( t \) such that the value is 1 if the corresponding entity is alive and 0 otherwise.

Based on the minimal cut set method, the system consists of six parallel subsystems (1,2), (1,3), (1,4), (2,3), (2,4) and (3,4) connected in series. Therefore,

\[
\phi(t) = [1 - (1 - X_1)(1 - X_2)] [1 - (1 - X_1)(1 - X_3)] [1 - (1 - X_1)(1 - X_4)] [1 - (1 - X_2)(1 - X_3)] [1 - (1 - X_2)(1 - X_4)] [1 - (1 - X_3)(1 - X_4)]
\]

\[
= (X_1 + X_2 - X_1X_2) (X_1 + X_3 - X_1X_3) (X_1 + X_4 - X_1X_4) (X_2 + X_3 - X_2X_3) (X_2 + X_4 - X_2X_4) (X_3 + X_4 - X_3X_4)
\]

\[
= (X_1 + X_2X_3 - X_1X_2X_3) (X_1X_2 + X_1X_3 + X_1X_2X_3 + X_2X_4 + X_3X_4 - X_2X_3X_4 - X_1X_2X_4 - X_1X_3X_4 + X_1X_2X_3X_4)
\]

\[
= (X_1 + X_2X_3 - X_1X_2X_3) (X_2X_3 - X_2X_3X_4 + X_4)
\]

\[
= (X_1 + X_2X_3 - X_1X_2X_3) (X_2X_3 + X_3X_4 + X_1X_2X_3 - X_2X_3X_4 - X_1X_2X_4)
\]

\[
= (X_1X_2X_3 + X_1X_2X_4 + X_1X_3X_4 + X_2X_3X_4 - 3X_1X_2X_3X_4)
\]

Consequently,

\[
E[\phi(t)] = R(t)
\]

\[
= E[X_1X_2X_3] + E[X_1X_2X_4] + E[X_1X_3X_4] + E[X_2X_3X_4] - 3E[X_1X_2X_3X_4]
\]

\[
\]

\[
= R_1(t)R_2(t)R_3(t) + R_1(t)R_2(t)R_4(t) + R_1(t)R_3(t)R_4(t) + R_2(t)R_3(t)R_4(t) - 3R_1(t)R_2(t)R_3(t)R_4(t)
\]

\[
= e^{-(\lambda_1+\lambda_2+\lambda_3)t} + e^{-(\lambda_1+\lambda_2+\lambda_4)t} + e^{-(\lambda_1+\lambda_3+\lambda_4)t} + e^{-(\lambda_2+\lambda_3+\lambda_4)t} - 3e^{-(\lambda_1+\lambda_2+\lambda_3+\lambda_4)t}
\]
3. (15 points.) Consider a fully-connected network topology shown above with 3 nodes (1, 2 and 3) and 3 links (a, b, and c). Two nodes can communicate with each other as long as there exists a communication path between them. For example, nodes 1 and 2 can communicate with each other via the following two communication paths: (a) link a only; and (b) a path consisting of links c and b passing through node 3. The system is designed such that a failed node can be bypassed without blocking any communication path in which it is an intermediate node. For example, if link a and node 3 fail, then nodes 1 and 2 can still communicate with each other by the path consisting of links c and b bypassing the failed node 3.

Assume that all nodes (all links) are indistinguishable. The failure and repair rates of each node are $\lambda_n = 0.0001 \text{ hr}^{-1}$ and $\mu_n = 0.001 \text{ hr}^{-1}$, respectively, while those for each link are $\lambda_l = 0.00005 \text{ hr}^{-1}$ and $\mu_l = 0.0005 \text{ hr}^{-1}$, respectively. The system requires that at least two nodes must be alive and be able to communicate with each other for the system to be operational. Use a fault tree model to compute the availability of the system at $t=5000$ hours.

(a) (10 points.) Show your fault tree model.
(b) (5 points.) Write a Sharpe code based on your fault tree model to compute the system availability.

Ans (a): The fault tree model below is based on the minimal path set method to deal with repeated components. An alternative solution is based on minimal cut set.

Ans (b):

```
poly abar(lambda,mu) gen
lambda/(lambda+mu),0,0
-lambda/(lambda+mu),0,-(lambda+mu)

ftree 3ring
* nodes are labeled with 1 2 3 and links are labeled with a, b and c
repeat 1 abar(lambdan,mun)
repeat 2 abar(lambdan,mun)
repeat 3 abar(lambdan,mun)
```
repeat a abar(lambda_d,mul)
repeat b abar(lambda_d,mul)
repeat c abar(lambda_d,mul)
*enumerating all minimal path sets in the 3-ring topology
or path1 1 2 a
or path2 2 3 b
or path3 3 1 c
or path4 1 2 b c
or path5 2 3 a c
or path6 3 1 a b
and top path1 path2 path3 path4 path5 path6
end

bind
lambda_n 0.0001
mun 0.001
lambda_d 0.00005
mul 0.0005
end

*print the availability at time=5000hr
expr 1-value(5000;3ring)
end

Output:
1-value(5000;3ring): 9.7331e-01