Solution to Homework #1
CS5214 Modeling and Evaluation  Fall 2023

1. (1) The customer turned away probability is 0.279878.
(2) The average number of customers waiting in the system is 4.786998.
(3) The response time per client for those served by the fast server is 0.642219 s.
(4) The response time per client for those served by the two slow servers is 0.7777516 s.
(5) The throughput is 10.798395
(6) The report confidence accuracy is H/Y = 0.003165/0.642219 = 4.9%.

Source Code:

```c
#include"smpl.h"
define TOKENS 1000
define TRUE 1
define FALSE 0

main()
{
    real Ta=1.0/15,Ts_fast=0.2,Ts_slow=1.0/3,mean,hw,nq;
    int tk_id=0,customer=0,event,mn310,nb;
    int n_fastServed=0,n_slowServed=0; /*number of customers severd by fast server or slow servers*/
    real R_sum_fast=0.0,R_sum_slow=0.0; /*sum of response times served by fast server or slow servers*/
    real n_arrived=0; /* number of customers arrived at the system*/
    int n_rejected=0; /*number of customers turned away by the system*/
    int n_waiting; /*number of tokens waiting in queue*/
    int fastBusy=FALSE,slow1Busy=FALSE,slow2Busy=FALSE; /*server busy or not */
    real ts[TOKENS];
    int cont=TRUE;
    smpl(0, "M/M/3/10 Queue with BMA");
    init_bm(200,20000);
    mn310=facility("mn310",3);
    schedule(1.0,0.0,tk_id);
    while (cont)
    {
        cause(&event,&customer);
        switch(event)
        {
            case 1:"arrival*/
            n_arrived++;
            if(++tk_id>=TOKENS) tk_id=0;
            schedule(1,expntl(Ta),tk_id);
            n_waiting=inq(mn310);
            if (n_waiting>=7) n_rejected++;
```
else{
    schedule(2,0.0,customer);
    ts[customer]=time();
}
break;
case 2;/*request server*/
if (request(mm310,customer,0)==0){
    if (!fastBusy){
        schedule(3,expntl(Ts_fast),customer);
        fastBusy=TRUE;
    }else if (!slow1Busy){
        schedule(4,expntl(Ts_slow),customer);
        slow1Busy=TRUE;
    }else{
        schedule(5,expntl(Ts_slow),customer);
        slow2Busy=TRUE;
    }
}
break;
case 3;/*departure from the fast server*/
release(mm310,customer);
fastBusy=FALSE;
n_fastServed++;
R_sum_fast+=time()-ts[customer];
if(obs(time()-ts[customer])==1)
    cont=FALSE;
break;
case 4;/*departure from the first slow server*/
release(mm310,customer);
slow1Busy=FALSE;
n_slowServed++;
R_sum_slow+=time()-ts[customer];
break;
case 5;/*departure from the second slow server*/
release(mm310,customer);
slow2Busy=FALSE;
n_slowServed++;
R_sum_slow+=time()-ts[customer];
break;
}
/*end while*/
civals(&mean,&hw,&nb);
/*answer for Q1*/
printf("Customer turned away probability: %fn", n_rejected/n_arrived);
/*answer for Q2*/
printf("Average number of customers waiting in the system: %fn", Lq(mm310));
/*answer for Q3*/
printf("Response time per client served by fast server: %f\n", mean);
/*answer for Q4*/
printf("Response time per client served by slow servers: %f\n", R_sum_slow/n_slowServed);
/*answer for Q5*/
printf("Throughput: %f\n", (n_fastServed + n_slowServed)/time());
/*answer for Q6*/
printf("Y=%f,H=%f after %d batches\n",mean,hw,nb);
}

Output:

batch  1 mean = 0.646
batch  2 mean = 0.641
batch  3 mean = 0.647
batch  4 mean = 0.638
batch  5 mean = 0.646
batch  6 mean = 0.643
batch  7 mean = 0.633
batch  8 mean = 0.639
batch  9 mean = 0.644
batch 10 mean = 0.644, rel. HW = 0.005
Customer turned away probability: 0.279878
Average number of customers waiting in the system: 4.786998
Response time per client served by fast server: 0.642219
Response time per client served by slow servers: 0.777516
Throughput: 10.798395
Y=0.642219,H=0.003165 after 10 batches
2. The system structure is shown as follows:

\[ R(t) = R_w(t)[1 - F_{f1}(t)F_{f2}(t)F_{f3}(t)] = R_w(t)[1 - (1 - R_f(t))^3] \]

\[ = e^{-\lambda_{\omega}t} \left[ 1 - (1 - e^{-\lambda_{f}t})^3 \right] = 3e^{-\lambda_{\omega}t - \lambda_{f}t} - 3e^{-\lambda_{\omega}t - 2\lambda_{f}t} + e^{-\lambda_{\omega}t - 3\lambda_{f}t} \]

MTTF = \( \int_0^\infty R(t)dt = \frac{3}{\lambda_{\omega} + \lambda_{f}} - \frac{3}{\lambda_{\omega} + 2\lambda_{f}} + \frac{1}{\lambda_{\omega} + 3\lambda_{f}} \)
3.

(a) The vertices of the reliability graph model are marked with a, b, c, d, and e as follows:

![Reliability Graph Diagram]

Source Code:

```plaintext
bind
r1  0.00001
r2  0.00002
r3  0.00003
r4  0.00004
r5  0.00005
r6  0.00006
r7  0.00007
end

relgraph bridge
  a b exp(r1)
  a c exp(r3)
  b d exp(r4)
  d e exp(r6)
  c e exp(r7)
end

bidirect
  b c exp(r2)
  c d exp(r5)
end

expr 1-value(2016;bridge)
pqcdf (bridge)
end
```

Output:

The system reliability after 12 weeks of operation is 0.98279.

(b) There are 7 minimal path sets:

\{1,2,5,6\}, \{1,2,7\}, \{1,4,5,7\}, \{1,4,6\}, \{2,3,4,6\}, \{3,5,6\}, \{3,7\}

There are 6 minimal cut sets:

\{1,3\}, \{6,7\}, \{2,3,4\}, \{4,5,7\}, \{1,2,5,7\}, \{2,3,5,6\}

The structure using a parallel connection of the minimal path sets is shown as follows:
The structure using a series connection of the **minimal cut sets** is shown as follows:

(c) The fault tree model built based on the **minimal path sets** is shown as follows:
The system reliability after 12 weeks of operation is 0.98279.