STATISTICAL COMPUTATIONAL METHODS

Seminar Nr. 6, Queuing Systems

Simulations

5. Messages arrive at an electronic mail server according to a Poisson process with the average frequency of 5 messages per minute. The server can process only one message at a time and messages are processed on a "first come $-$ first serve" basis. It takes an Exponential amount of time M_1 to process any text message, plus an Exponential amount of time M_2, independent of M_1, to process attachments (if there are any), with $E(M_1) = 2$ seconds and $E(M_2) = 7$ seconds. Forty percent of messages contain attachments. Use Monte Carlo methods to estimate
a) the expected response time of this server;
b) the expected waiting time of a message before it is processed.

Solution:

The process of arrivals is Poisson, so the interarrival times are Exponential with an average of 5 messages per minute (i.e. $\mu_A = 5$ minutes). Notice that because of the attachments, the overall service time is *not* Exponential, so this system is *not* M / M / 1, it is M / G / 1, where "G" stands for "general". For service times, we have $\mu_{M_1} = 2$ seconds and $\mu_{M_2} = 7$ seconds.
For each job,

$$\begin{aligned} \text{response time} &= \text{departure time} - \text{arrival time}, \\ \text{waiting time} &= \text{service starting time} - \text{arrival time}, \end{aligned}$$

so we will keep track of arrival times, of times when service starts and times when service finishes (departure times), as arrays. In addition, we need to know when the server becomes available, so we will have a variable for that time. We start with the parameters and initialization of variables

```
% Generate an M/G/1 queuing system.
clear all

% t = cputime; % if you want to compute the CPU time
% Parameters of the system
lamA = 5/60; % arrival rate lambdaA, per sec.
lamM1 = 1/2; % parameter of M1 (texts); lambdaM1
```

```
lamM2= 1/7; % parameter of M2 (attachments); lambdaM2
p = 0.4; % proportion of emails with attachments
N = input('size of MC study = '); % size of the MC study:
                                  % number of generated jobs

% Initialize variables
arrival = zeros(1, N); % arrival times
start = zeros(1, N); % times when service starts
finish = zeros(1, N); % times when service finishes;
                      % departure times
T = 0; % arrival time of a new job
A = 0; % time when the server becomes available
```

Then each new arrival time is an $Exp(\lambda_A)$ variable. For service time, it is $M_1 \in Exp(\lambda_{M_1})$ plus the service time for attachments $M_2 \in Exp(\lambda_{M_2})$, with probability $p = 0.4$. Whether or not we add the time for processing attachments is a Bernoulli variable with parameter $p = 0.4$,

$$\begin{pmatrix} 0 & 1 \\ 1-p & p \end{pmatrix}.$$

Remember that this is simulated as $U < p$ ("success"), for a Standard Uniform variable U. So the *total* service time will be $M_1 + (U < p) * M_2$. For each job, the service starts either when the job arrives or when the server becomes available, whichever happens *last*.

```
for j = 1 : N % do-loop over N jobs
    T = T - 1/lamA * log(rand); % arrival time of the j-th job
    S = -1/lamM1 * log(rand) - (rand < p) * 1/lamM2 * log(rand);
                      % service time of the j-th job
    arrival(j) = T; % arrival time of the j-th job
    start(j) = max(A, T); % time when service starts
    finish(j) = start(j) + S; % departure time
    A = finish(j); % time when the server becomes available
                   % to take the (j+1)st job
end
% el_time = cputime - t; % elapsed time, if you want
                         % to compute the CPU time
```

Then we summarize the results.

```
% Output
fprintf('a) expected response time E(R) is %3.5f sec.\n', ...
            mean(finish - arrival))
fprintf('b) expected waiting time E(W) is %3.5f sec.\n', ...
            mean(start - arrival))
% fprintf('CPU time = %f seconds\n', el_time)
```

Run it a few times with $N = 1e5, 5e5, 1e6$. We can estimate the expected response time as ≈ 8.9 seconds and the expected waiting time ≈ 4 seconds.

6. A small clinic has several doctors on duty, but only one patient is seen at a time. Patients are scheduled to arrive at equal 15-minute intervals, are then served in the order of their arrivals and each of them needs a Gamma time with the doctor, that has parameters $\alpha = 4$ and $\lambda = 10/3 \text{ min}^{-1}$. Use Monte Carlo simulations to estimate
a) the probability that a patient has to wait before seeing the doctor;
b) the expected waiting time for a patient;

Solution:

This system is *not* M/M/1 either, both because the interarrival times are not random and because the service times are not Exponential. This is a D/Ga/1 queuing system, where "D" stands for "deterministic" (fixed) and "Ga" for Gamma.

Now arrival times are fixed, at increments of 15 minutes. Service times have a $Gamma(\alpha, \lambda)$ distribution, which can be simulated as a sum of α $Exp(1/\lambda)$ variables. Let us notice that the mean interarrival time is $\mu_A = 15$ minutes, while the average service time is $\mu_S = \alpha \cdot \lambda = 40/3 = 13.33$ (the expected value of a $Gamma(\alpha, \lambda)$ variable), so the system is functional $r = \mu_S/\mu_A < 1$.

Let us remember that the range of values of a random variable is mean $\pm$ 3 st. deviations (consequence of Chebyshev's inequality). In this case, the standard deviation of service times is $\sqrt{\alpha \cdot \lambda^2} = \dfrac{20}{3} \approx 6.67$, quite large, so we can expect quite a bit of variation.
Again, we keep track of arrival times, of times when service starts, times when service finishes (departure times), as arrays and time when the server becomes available.

```
% Generate a  D/Ga/1 queuing system.
clear all

tcpu = cputime; % if you want to compute the CPU time
```

```
% Parameters of the system
alpha = 4; lambda = 10/3; % parameters of service times
t = 15; % fixed interarrival times
N = input('size of MC study = '); % size of the MC study:
                                  % number of generated jobs
arrival = 0 : t : (N - 1) * t; % arrival times = 0, t, 2t, 3t, ...

% Initialize variables
start = zeros(1, N); % times when service starts
finish = zeros(1, N); % times when service finishes;
                      % departure times
A = 0; % time when the doctor (server) becomes available

for j = 1 : N % do-loop over N jobs
    start(j) = max(A, arrival(j)); % time when service starts
    S = -lambda * sum(log(rand(alpha,1)));
      % service time for each job, Gamma(alpha, lambda) distr.
    finish(j) = start(j) + S; % departure time
    A = finish(j); % time when the doctor (server) becomes
                % available to take the (j+1)st job (patient)
end
el_time = cputime - tcpu; % elapsed time, if you want
                          % to compute the CPU time

% Output
fprintf('a) prob. that a patient has to wait P(W > 0) ...
         is %3.5f\n', mean(start > arrival))
fprintf('b) expected waiting time E(W) is %3.5f min.\n', ...
         mean(start - arrival))
fprintf('CPU time = %f seconds\n', el_time)
```

After several runs (try $N = 1e5, 5e5, 1e6$), we find the probability that a patient has to wait to be ≈ 0.66 and the average waiting time for a patient between 10 and 11 minutes.