Dept. of Computer Science and Engineering,

CS69011: Computing Lab 1

Assignment 3

Maximum Marks: 30

Complex dynamical systems are studied using a technique called *discrete event simulation* (also called *event driven simulation*). For a high level understanding of this technique, please see:
https://en.wikipedia.org/wiki/Discrete-event_simulation

A suggested site for some interesting exercise problems in discrete event simulation is the one below:
https://algs4.cs.princeton.edu/61event/

AN EXAMPLE TO DESCRIBE THE WORKING OF A DES

Consider a stream of tasks executing in two processors, P_1 and P_2. The arrival rate of the tasks follow a Poisson arrival process with mean arrival rate of $1 / \lambda_a$. On arrival a task is allocated to P_1 with a probability of 0.4 and to P_2 with a probability of 0.6. Each processor executes the tasks allocated to it sequentially one after the other. The execution time of the tasks follow the exponential distribution with mean λ_b. We wish to simulate the system and compute various features of the system, such as task throughput (number of tasks completed per second), average load on each processor (number of tasks pending in its queue), and expected waiting time (time for which a task waits before being executed).

In an event driven simulator, we maintain a queue of events ordered by time stamps. In this case, we will use two types of events, namely *arrival events* for the arrival of the tasks one after the other (separated by time), and *service events* (representing the completion of tasks). Broadly, we do the following:

1. In the beginning, we have only the arrival event, $\langle E_a, t \rangle$, for the first task. In general, every arrival event, $\langle E_a, t \rangle$, tells us that the next task will arrive at time t.
2. In each iteration, simulation time progresses to the time of the first event, that is, the time stamp of the first event in the queue.
3. When an arrival event with timestamp, t, is processed to create the next task, T_j, we do the following:
   (a) We choose the processor, X, (where X=1 or X=2) for executing the task. The choice of X (1 or 2) is made randomly, such that P[X=1] = 0.4 and P[X=2] = 0.6.
   (b) We add a service event, $\langle E_s, X, t + \delta \rangle$, for that task into the queue. The execution time, δ, for the task is chosen randomly following the exponential distribution. This will be explained shortly.
   (c) We also add another arrival event, $\langle E_a, t + \Delta \rangle$, into the queue. This models the arrival of the next task, where the inter-arrival time, Δ, is chosen as per the Poisson arrival process. This will be explained shortly.
4. When a service event is processed, we mark the task as completed and note the completion time.

Note that time is incremented in non-uniform amounts, that is, the time between successive events. This makes event driven simulators much faster than simulators which increase time by a constant amount.

We will need random number generators here. Please look up the library functions, `rand()`, `srand()`, and `drand48()`.

Let us first understand how we choose the processor to which a new task will be allocated. We are given that P[X=1] = 0.4 and P[X=2] = 0.6. By default, we assume that the decision follows the uniform distribution. To simulate the choice between X=1 and X=2, we use the random number generator to choose a random number between 0 and 1. If the chosen number is less than 04, we choose X=1, otherwise we choose X=2.

It is a little more tricky to choose the interval times, Δ, and the service times, δ, because these do not follow the uniform distribution. When arrival rates follow the Poisson distribution, the inter-arrival times follow the Exponential distribution (this is a well-known result in mathematics). Exponential distribution has a *memoryless property* which makes it a natural choice for modeling arrival and service processes. The memoryless property means that any of the value of the random variable is independent of the previous choices. In our context, if execution times are exponentially distributed, then the service time of each task is independent of the service times of all previous tasks. Likewise, if inter-arrival times are exponentially distributed, then each inter-arrival time is independent of all previous inter-arrival times.

The *probability density function* of the exponential distribution is:

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

The *cumulative density function* is therefore:

$$F(t) = \int_{-\infty}^{t} f(x)dx = \begin{cases} 1 - e^{-\lambda t}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

From this we get:

$$t = \frac{-\ln(1 - F(t))}{\lambda}$$

Here, $F(t)$, and therefore $1 - F(t)$, are uniform random. Therefore to choose the inter-arrival time randomly following an exponential distribution of mean λ we may use the formula, $t = \frac{-\ln(r)}{\lambda}$, where *r* is a random number between 0 and 1, generated using a uniform random generator like `rand()` or `drand48()`.

THE PROBLEM STATEMENT

The aim of this assignment is to learn how to develop an event-driven simulator for studying a dynamical system. You are required to develop the simulator for the following system:

The BCRT Hospital facility is planning to develop a cardiac check-up unit with five test facilities, namely ECG, Echocardiogram, Treadmill, MRI, and CT. The following table shows the packages offered, the registration timings, and the mean inter-arrival times of patients arriving under each package.

Package	Tests	Registration time	Mean inter-arrival time
P1	ECG, Echo, Treadmill	6:00 – 11:00	λ_1
P2	MRI	8:00 – 15:00	λ_2
P3	CT	8:00 – 15:00	λ_2
P4	ECG	6:00 – 17:00	λ_3 between 6:00 – 12:00, and λ_4 after 12:00
P5	Echo	6:00 – 15:00	λ_4
P6	ECG, Echo	6:00 – 15:00	λ_5

The mean service times for the registration and the different test facilities are given in the following table.

Test Facility	Mean service time
Registration	5 mins
ECG	10 mins
Echocardiogram	20 mins
Treadmill	40 mins
MRI	40 mins
CT	30 mins

Both inter-arrival times and service times follow the exponential distribution.

Task-1:
Develop an event driven simulator for the above system. You must model the arrival process for patients for each package and the service processes of each test facility, including the registration. The registration process is common for all packages. Thereafter the registration counter informs the patient the sequence in which they must visit the various test facilities for which they registered.

For this task, let us assume that this sequence is the same for all patients in a package. For example, for package P1, the sequence may be ECG → Echo → Treadmill; and for P6, the sequence may be Echo → ECG. The patient will visit the next facility in her package after completing each test in her package. Note that the service times are random, and therefore exact times of arrival for the patient at each facility cannot be predicted a priori.
[Hint: *Do not generate the service time for a patient in a test facility before she has completed the previous tests – because this will need a lot of housekeeping – think* !! *A better option is to generate an arrival event at the next test facility at the same time as the service completion event of the previous test.*]

Compute the following statistics from your simulation (*do not store all the simulation data*) of the facility over at least 10 days:

(a) The average total time spent by patients for each package.
(b) The test facility with the highest and lowest average utilization.
(c) The average time of the day at which each facility closes. Note that a facility must remain open until all patients registered for that test have been served.
(d) Can we improve the average total time spent by all patients by changing the sequence of tests in the packages?
(e) Which facility may we duplicate to get maximum improvement in throughput? Throughput is defined as the number of patients served per unit time, where a patient is served when all her tests are complete.

Task-2:
Model and evaluate separately each of the following policies with respect to the statistics described in (a), (b), and (c) in Task-1:

(a) The hospital will prioritize patients in the test facilities: P1 > P6 > P5, P4
(b) The hospital will dynamically choose the next test facility for a patient from her package, so that the expected waiting time in the next facility is minimum.