CA4011 (2018-2019)
Continuous Assignment Specification

Part A (Simulation)

CONTENTS

1 Overview of continuous assessment

1.1 Marks breakdown

Part A of CA4011 continuous assessment involves simulation. Part B (*to be specified later in a separate document*) will be on Mathematical Programming. The continuous assessment part of this module is worth 30% of the overall module marks and these will be shared equally between Parts A and B. Thus,

Part A is worth 50% of the CA mark or 15% of the overall module mark.

Part B is worth 50% of the CA mark or 15% of the overall module mark.

1.2 Deadlines

The deadlines for submission of the work (through LOOP) are as follows:

Part A: 23.00 on Friday March 15th, 2019

Part B: 23.00 on Thursday April 18th, 2019

1.3 Performance of Part A

This part of continuous assessment ***may be done individually or in pairs***. ***Please inform*** the lecturer by email (one per pair!) ***by Friday February 15th*** (2019) whether you are doing the assessment as an individual or with a partner. If the latter, specify your partner's name and student ID.

The main deliverable (in pdf format) is to be a report containing a concise description of the design or structure of your software simulation model(s), preferably with some diagrams. This should be followed by the results you obtained for each of the elements and sub-tasks (see Section 2, below). The report should finish with short conclusion section(s). The source code (the choice of language is up to you) should be included in an appendix.

2 Specification of Part A

2.0 Introduction

Part A is made up of three sub-elements with the total marks for part A distributed as follows:

	One person team	**Two person team**
Element A1	10%	5%
Element A2	90%	60%
Element A3	***Not applicable***	35%

2.1 Element A1: Random Number Generator Test (One & Two person teams)

All students must complete this element. This is a specific piece of work intended to increase insight into (pseudo) random number generators. Section 1.4.6 "Generating pseudo-random numbers for Uniform distribution" of the lecture notes sketched some of the ideas and presented a few example generator formulae *(Fibanocci* and *Multiplicative Congruential*). Also, as mentioned in the notes, there are various **statistical tests** that can be applied to test the "randomness" of a particular generator.

This piece of the continuous assessment work is to apply a particular (simple!) statistical test, called the Chi-squared test, to

(A) the *Multiplicative Congruential* method to see how well it performs for various values of its parameters (u_0, a, b, m). *The choice of parameter values is up to you*.

(**B**) a uniform random number generator in a programming language or software tool of your choice (e.g. Python, C, Java, Matlab, R, …). This will serve as a reference for **(A)**.

Definition of the Chi-squared test: The idea of this test is to check whether or not the numbers produced are spread out reasonably. If we generate N non-negative integers in the range [0, r-1] then we would expect to get about N/r occurrences of each value. However, the frequency of occurrence should not be exactly the same as that would not be random. The Chi-squared test involves calculating the statistic $\frac{r}{N}\sum_{i=0}^{r-1}(f_i - \frac{N}{r})^2$, where f_i is the frequency of occurrence of integer i. If the statistic is close to r then we conclude that the numbers are random, otherwise not. Specifically, for $N > 2r$, it has been proved that the Chi-squared statistic value should be within $2\sqrt{r}$ of r.

<u>Explanatory example</u>: The function RandomInteger[r] in ***Mathematica*** generates a (pseudo) uniform random integer in the range [0, r-1]. For example, if N = 1000 and r = 10, one "run" found the following frequencies of occurrence using that function: {{0,115}, {1,103}, {2,105}, {3,97}, {4,103}, {5,104}, {6,99}, {7,86}, {8,98}, {9,90}}; for example, there were 115 occurrences of 0. Twenty repetitions of such a run resulted in the following values of |Chi-squared statistic – r|: {1.2, 2.12, 1.3, 7.7, 1.42, 4.24, 7., 1.84, 4.74, 3.04, 2.28, 2.88, 1.46, 5.58, 3.38, 4.78, 6.08, 2.74, 0.92, 1.22}. Here, 2√r = 6.3 (approx) so, as the values we found for all twenty replications were less than 6.3, we conclude that the values generated are random.

2.2 Element A2: Hospital outpatients (One & Two person teams)

2.2.0 Introduction to element A2

The objective is to build a software simulation model of a system to which customers arrive to receive a service, and then to use the model to investigate a number of scenarios. There may be one or more servers to provide the service. The purpose of the model is to evaluate the performance of the system as the patterns of arrivals and services are varied.

In the following, the particular system described is a hospital outpatient clinic but the model should be applicable to other analogous systems. For the hospital outpatient system, the customers are patients (or clients) and the servers are health care professionals (usually doctors but, depending on the clinic, might be nurse specialists, phlebotomists, etc). The overall goal is to model the system for a typical session; in the following, an example is considered where a session is one day in which the clinic is open for patient arrivals between 09.00 and 17.30 (but your software should allow these and other parameters to be varied).

The task is broken into a number of sub-tasks of increasing complexity. However, the software should be structured as a unified system where users can specify input parameter values to define the different scenarios to be modelled.

In stochastic simulations (i.e. those involving some random elements) it is essential to provide for ***replications*** of experiments. This is necessary in order to provide a measure of the confidence that may be placed on the results obtained. Typically, there might be 100 or more replications and, from these, averages and standard errors can be calculated as explained in lecture notes (see section 1.2 of the notes on simulation, for example). The ***number of replications to be applied should be an input parameter*** for the simulation system.

In queueing systems, there are ***various measures of performance*** including

Average time (W) a customer spends in the system (queueing & being served)

Average time (W_q) a customer spends waiting for service (that is, in the queue)

Maximum time a customer spends in the system (queueing & being served)

Maximum time a customer spends waiting for service (that is, in the queue)

Proportion of time each server is idle (P_0)

Average number (L) of customers in the system (queueing & being served)

Average number (L_q) of customers in the queue

Your system should ***calculate and output these measures of performance***. An average value and a standard error should be should be reported for each measure of performance (e.g. see courses notes Part C, Section 1.2).

Part of the motivation for this assignment comes from (Teow, 2009) where some Operations Research applications are highlighted. This short paper is reproduced in LOOP and should be read as background, particularly "Application 3: Queue design for stochastic demand" which suggested Task 2 of this assignment, and ""Application 2: Outpatient Appointment Scheduling" which suggested Task 4 of this assignment.

In all the following tasks, the distribution of service times should be assumed to be exponential. However, the distribution of arrivals varies according to the particular task.

2.2.1 Task 1: Compare random and scheduled patterns of customer arrival

This task builds on Section 1.2 of the lecture notes on simulation so familiarising with that section is a good point to start from. Another source that might be useful is Section 4.8 of (Matloff, 2009) which contains an example of an R program for a discrete event simulation.

<u>Services</u>: As in Section 1.2 of the notes, assume there are two (*but your model should allow for more*) servers (health care professionals) who are identical in terms of how quickly they treat patients. In the case of doctors, for example, you might assume plausibly that each doctor can see 4 patients in one hour so that the service rate $\mu = 4$. On the other hand, if the servers were phlebotomists a server rate of 10 might be more realistic.

<u>Arrivals</u>: In section 1.2 of the notes, a **scheduled pattern of arrivals** was considered:

> Arrivals at times 1, 2, … but with some uncertainty around the arrival time

To begin with[1], for a medical clinic (with doctors as servers), you might assume arrivals scheduled at 09.05, 09.15, 09.25, … ,17.25 with the actual arrival points independently distributed around the scheduled arrival times points as mean and with standard deviation 5 minutes. In this case, the average arrival rate $\lambda = 6$ per hour.

Note: Following section 1.2 of the notes, these sample data for the case of doctors as servers give $\lambda/\mu = 6/4 = 3/2$ so that traffic intensity $\rho = \lambda/(2\mu) = ¾$.

[1] ***NB: However, your software should provide for these and other constants in this specification to be input parameters to allow for other values to be used.***

(a) The first part of this task is to carry out the simulation, essentially following the approach of Section 1.2 of the notes. The output from the simulation should be estimates (averages and standard errors) of the various performance parameters listed in section 2.0 (above). In addition, it would be of interest to estimate when the last patient leaves.

(b) The second part of the task is to replace the pattern of arrivals by a random pattern where the arrivals are Poisson distributed (equivalent to the inter-arrival times being exponentially distributed) with parameter λ. The service times are unchanged. Then the simulation should be performed again leading to estimates of the various performance parameters.

You should then compare the results of (a) and (b) and draw conclusions as to which system works better. In the case of (b) you should also compare your simulation results to the theoretical values predicted by Queue Theory for W, W_q, L, L_q and P_0.You should present results for three different sets of λ and μ values to check whether the conclusions are the same for each set.

2.2.2 Task 2: Effect of separate queue for each gender

This task builds on the work of Task 1.

It is required to examine what the impact would be on the system performance measures if one server sees female patients only and the other server sees male patients only. You may assume and build into your software that the probability is that same (0.5) of a patient being male or female.

For this task, you should apply both types of arrival pattern specified in Task 1 ((a) scheduled arrival pattern and (b) random pattern) and present a comparison of your findings with those of Task 1.

As for Task 1, you should present results for three different sets of λ and μ values to check whether the conclusions are the same for each set.

2.2.3 Task 3: More realistic pattern of service (with scheduled arrivals)

This task also builds on the work of Task 1 and, for this task, it is sufficient to use only the scheduled arrival pattern (a) specified for Task 1. As in Task 1, you should assume two identical servers.

Here, it is proposed to vary the server availability to a more realistic pattern where each server (health professional) has a break both morning and afternoon. For example, to begin with, you might specify:

> Server 1 takes a break for 30 minutes at 10.45 and at 14.45, or whenever they finish with their current patient.
>
> Server 2 takes a break for 30 minutes at 11.15 and at 15.15, or whenever she/he
>
> finishes with his current patient.

The task then is to estimate the performance parameters and compare the performance of this somewhat more realistic system with that of (a) of Task 1.

As for Task 1, you should present results for three different sets of λ and μ values to check whether the conclusions are the same for each set.

2.2.4 Task 4: Customers not all the same – how best to manage?

In this task we assume that there are two different kinds of patient, regulars (or follow-up) patients and new patients. It is to be expected that the service times for new patients are longer than for regular patients (though still exponentially distributed). For example, we might have that the average consultation time for a regular patient is 15 minutes but that for a new patient is 30 minutes.

It is also necessary to specify the proportion of regular and new patients – for example, one might have that 5/6 are regular and 1/6 are new.

(a) The first part of this task is repeat the simulation of Task 3 with the regular and new patients intermixed. For example, on average every 6th patient might be new. Then compare the results with those for Task 3. As usual, it is important to vary the parameter values. Thus, you should present results for at least three different sets of λ and μ values to check whether the conclusions are the same for each set. In addition, you should repeat this for varying proportions (e.g. 5/6:1/6, 2/3:1/3 and ½:½) of regular to new patients.

(b) The second part of the task is to perform simulations in which the regular patients are all scheduled before the new ones. The idea behind this is to try to prevent the problem of "snowballing" waiting times mentioned in (Teow, 2009, p. 565)**. Present a comparison of these simulations with those of the inter-mixed simulations (a).**

2.2.5 Task 5: Experience and Inexperience

This task builds on the work of Task 1, arrival pattern (a) only.

(i) It is required to examine what the impact would be on the system performance measures if there is only one server (doctor) who, however, works at double rate of the doctors in Task 1. For example, if the doctors in Task 1 see 4 patients per hour this faster (perhaps more experienced) doctor sees 8 patients per hour.

(ii) It is required to examine what the impact would be on the system performance measures if there are three servers (doctors) but two of them are very inexperienced. One doctor works at the same rate as those in Task 1 (for example, 4 patients per hour) but the other two work at half the rate (for example, 2 patients per hour).

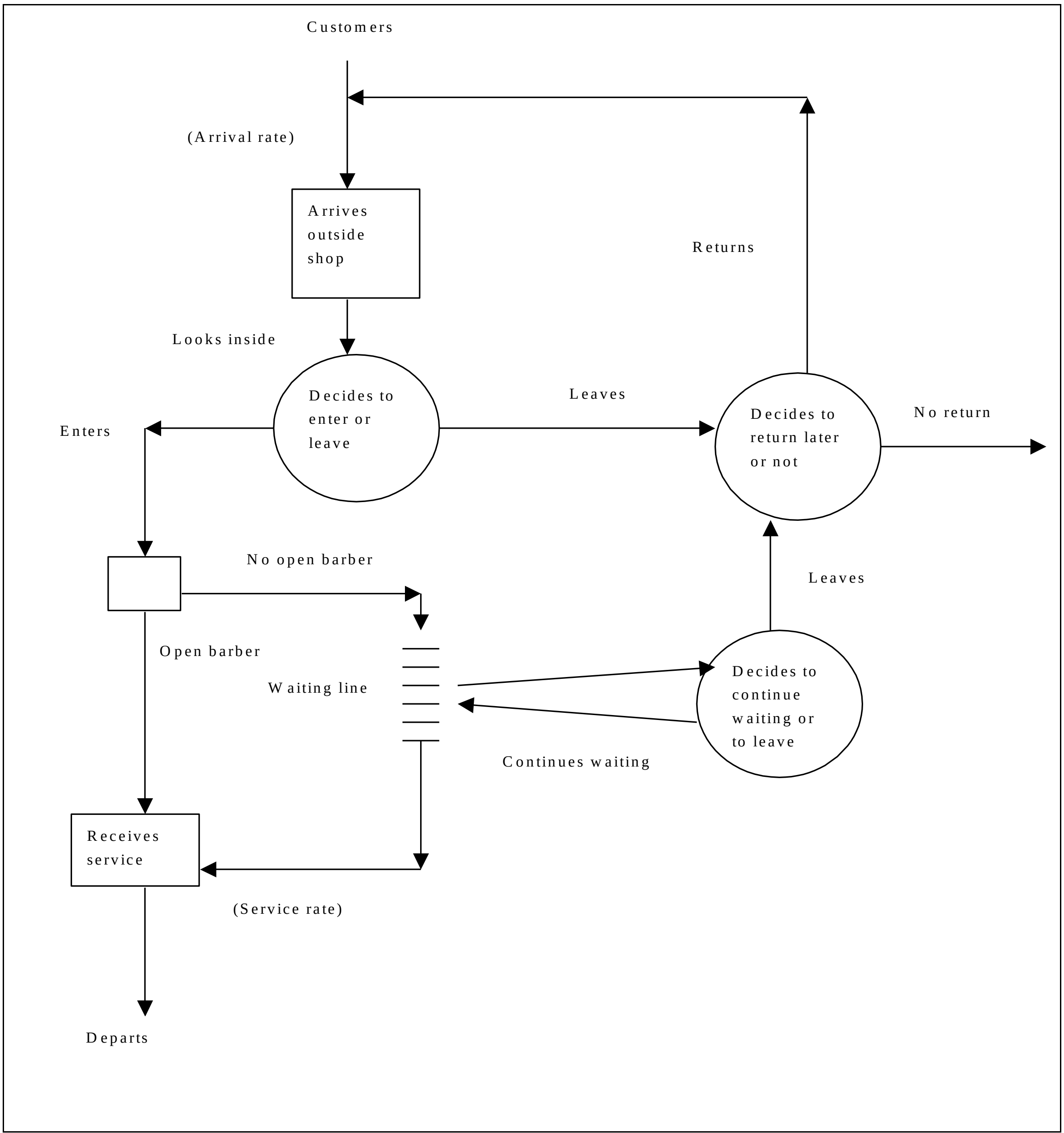
You should present a comparison of your findings for (i) and (ii) with those of Task 1(a) and with each other.

As for Task 1, you should present results for three different sets of λ and μ values to check whether the conclusions are the same for each set.

2.3 Element A3: Barbershop simulation (Two person teams only)

2.3.1 Simulation of male barbershop – 1 specific replication

Figure 1 depicts a queueing process that might occur at a barbershop:



<u>Figure 1: Barbershop queue</u> [*Circles represent points of decision for customers*]

Suppose that there are three barbers and that the time during which a customer is in the chair averages 20 minutes and is normally distributed with a standard deviation of 5 minutes. During the peak lunch-hour period from 12.00 to 13.45 customers arrive in a Poisson stream with an average of 12 per hour (that is, the inter-arrival durations (or times) are exponentially distributed with parameter λ = 12). There are 3 chairs for waiting customers; when they are full (i.e. 6 customers in the shop in all), further arrivals leave immediately. If all barbers are busy and if there are, respectively, 0, 1, and 2 waiting customers, the probabilities that an arrival leaves without waiting are 0.1, 0.3 and 0.5. Customers who have waited 15 minutes and are not about to receive service have an even chance of leaving without waiting further.

From the opening of the shop (10.00) to 12.00 and from 13.45 to closing time (17.00), customer arrival is at the rate of 6 per hour (also a Poisson stream). One barber goes to lunch[2] from 11.00 to 11.30, and on that barber's return a second barber goes from 11.30 to 12.00. The third barber takes lunch from 14.00 to 14.30.

<u>What is required?</u>

- You are required to develop a software model to simulate this system.
- The main purpose of the simulation is to answer the question,

"***On an average day, how many customers leave without receiving a haircut?***"

Note: The answer to this question would help in deciding whether to have a 4th barber.

- In addition, the software model should be able to output the detailed sequence of events for a typical day.

Assumptions: Your implementation may, if you find it convenient, use 5 minutes as the minimum time interval to be considered. In that case, assume that all events occur at the end of an integral number of 5-minute intervals.

NB: The foregoing description prescribes specific constant values for arrival rates, service rates etc. As is normal good practice, your implementation should be parameterised such that any change of value is easily accommodated.

2.3.2 Simulation of male barbershop – Many replications

It would be unwise to base a decision on whether to hire a 4th barber on a single day simulation. Therefore, your implementation should be capable of replicating the simulation for several days and, for each day, recording what proportion of customers leave without receiving a haircut. It should then calculate the average and standard deviation of the set of proportions and hence compute an estimate of the standard error. You should present results for 100, 200 and 400 replications.

[2] Assume that each barber completes the service for the customer in his or her chair when the barber's scheduled lunch period starts and then takes a full half hour for lunch. The second barber should not leave for lunch until the first barber has returned.

<u>Note (Teams of 2 only)</u>: It is acceptable to have, for elements A2 and A3, either two separate pieces of software or a single (flexible) piece of software.

Works Cited

Matloff, N. (2009). *The Art of R Programming.*

Teow, K. L. (2009). Practical Operations Research Applications for Healthcare Managers. *Annals Academy of Medicine, 38*, 564-566.