Practical Assignment 1

COMP40010 / COMP40550

Submission deadline: 23:59 on Friday 26th March 2021

Prof. Liam Murphy & Prof. John Murphy

Subject to change – Version 1.0

Each question concerns a particular type of queueing system, and the relevant mathematical formulas are taken from the lecture notes and provided with each question. You can use the excel sheets provided, or indeed any other calculators or code to get your results. The submission should be a short (no more than 10 pages!) report with an executive summary (a few sentences / paragraph) at the start of each section, and then a more technical discussion with graphs in the remainder of the section. Ideally you would submit this in a single pdf.

Question 1

You are designing a large enterprise computing system and you are examining the code that calls to a persistent storage database system. You have instrumented the code and determined that the time to service this call request is constant, at 85 milliseconds. You have also studied the number of requests arriving and observed that this varies with time in a random manner. The code only allows one access to the database at a time.

1. Plot the average number of requests (to two places of decimals) in the queue.

2. Also plot the average waiting times (rounded to milliseconds) for arrivals varying from ***one every 180 milliseconds*** to ***one every 90 milliseconds*** (in ten millisecond intervals).

Relevant mathematical equations for this M/D/1 queue (memoryless arrivals, deterministic service, single server and infinite waiting places).

M/D/1 **Number in the system $N=(\rho/(1-\rho))*(1-\rho/2)$**
Number in the queue $Nq=N-\rho$
Time spent in the system $T=(1/\mu\ (1-\rho))*(1-\rho/2)$
Time spent in the queue $W= T-(1/\mu)$

[Indicative Weight 10%]

Question 2

Suppose you change the data in the database and this lowers the average time to service the request. When you re-measure the time for service, you notice it varies considerably, and on average it takes 47 milliseconds.

1. Plot the average number of requests (to two places of decimals) in the queue.

2. Also plot the average waiting times for arrivals from ***one every 180 milliseconds*** to ***one every 60 milliseconds*** (in ten millisecond intervals).

3. As the arrival rate varies, there is a point where a request spends as long waiting in the queue as the time it takes to serve it (on average). What arrival rate is this, and what is the load at this critical point?

Relevant mathematical equations for this M/M/1 queue (memoryless arrivals, memoryless service, single server and infinite waiting places).

M/M/1	**$N=\rho/(1-\rho)$**
	$Nq=(\rho^2)/(1-\rho)$
	$T=(1/\mu)/(1-\rho)$
	$W= (1/\mu)\ \rho\ /(1-\rho)$

[Indicative Weight 10%]

Question 3

While you are happy with the overall changes, there is still a problem with how long the requests must wait. Suppose you decide to limit the number of requests that can wait in the queue to **n-1**. Now there will be requests rejected (referred to as "lost") but the ones that get in should have a shorter waiting time. You think that there is a trade-off between the loss rate (blocking probability) and the overall queueing time (W) rounded to milliseconds.

1. Plot the effect on the loss-versus-queueing-time trade-off as **n** varies. For example, pick many different values for **n**, and then vary the inter-arrival times from ***180 milliseconds*** to ***60 milliseconds***, in 20 milliseconds intervals.

2. Comment on what happens when the loading is greater than 1.

3. Explore what happens when you fix a load and then vary **n** to see how the loss rate and the overall queueing time are affected.

Relevant mathematical equations for this M/M/1/n queue (memoryless arrivals, memoryless service, single server and finite number of waiting places = n-1).

M/M/1/n	**Probability of Loss (or Blocking) $PL=(1-\rho)(\rho^n)/(1-(\rho^{(n+1)}))$**
	$N=\rho+(\rho n-n-1)(\rho^{(n+1)})/((1-(\rho^{(n+1)}))(1-\rho)$
	$T=N/\gamma$
	where $\gamma=\lambda(1-PL)$
	$W=T-(1/\mu)$

[Indicative Weight 10%]

Question 4

There is a major re-design of the architecture and the suggestion is that multiple simultaneous accesses to the database will be allowed. You are going to model the number of connections to be a maximum of **K**. The new architecture of this database logic means that the response time varies a lot, but on average it takes 26 milliseconds. You do not want to lose any of the requests so you are going to let them all queue up (none are "lost"). The chief enterprise architect thinks about having separate queues for each connection (like a supermarket queue), but this is a bit complex to implement. Instead, they decide to have a single queueing system for all requests, and when a connection is free to take a request, the request at the top of the queue moves to the database (like a bank queue).

You are asked to estimate how many requests will be in the queue (**Nq**) and how long they wait before they get service (**W**), for different numbers of possible connections. The initial guess is that you should support 12 connections, but the architect wants to see what happens when there are somewhere between 4 and 10 connections available (use this range of values for **K**). The latest estimates are that requests will have arrival rates of between 3 per second to about 23 per second, and you want to see how many connections you would need and the resulting delays and queue size for each scenario.

1. Use the range of arrival rates (3 per second to 23 per second) for all ten connections (**K**=10) and find the resulting delays and queue size.

2. Now for eight connections (**K**=8) you might only try the rates up to a maximum of 18 per second (to keep the queue stable).

3. Try other values for the number of parallel connections e.g. **K**=6, **K**=4.

4. Explain what happens when you double the number of connections and at the same time double the rate at which requests arrive. Investigate if you have the same results. Discuss your answer and how this result could lead to a general rule for resource usage.

Consider rounding the number of requests in the queue and the time to wait in milliseconds.

Relevant mathematical equations for this M/M/K queue (memoryless arrivals, memoryless service, K servers and infinite waiting places).

M/M/K	**PD= 1/ (K!((1- ρ/K)/(ρ^K))(Sum(i=0,K-1){ (ρ^i)/i!}+1))**	***Erlang C***
	Nq=PD*(ρ/K)/(1-ρ/K)	
	W=PD/(Kμ(1-ρ/K))	

[Indicative Weight 20%]

Question 5

Suppose you are working for an online bank that is about to launch a new service and you are monitoring the performance of the system. You have done some research and you think that about 210,000 customers will use the service per day, with a peak rate of between 5 and 25 per second. You have parallelised the application and you have it running on a Virtual Machine Cluster with 16 instances. You have optimised the database access to ensure that there is no locking between customers. Your initial testing of the system indicates that you are able to serve a customer request on average in 632 milliseconds of processor time.

1. Calculate the number of requests waiting and the average delay for the loads.

2. Determine the total load you would be able to handle if you wanted a customer to wait no more than one second.

3. You have an option to buy another cluster of 16 Virtual Machines. What is the load allowed now if you still do not want a customer to wait more than one second? How does this compare to the load with half the Virtual Machines?

4. How efficient is this if you are only getting arrivals of 25 per second and you have the 32 Virtual Machines?

Relevant mathematical equations for this M/M/K queue (memoryless arrivals, memoryless service, K servers and infinite waiting places).

M/M/K	**PD= 1/ (K!((1- ρ/K)/(ρ^K))(Sum(i=0,K-1){ (ρ^i)/i!}+1))**	***Erlang C***
	Nq=PD*(ρ/K)/(1-ρ/K)	
	W=PD/(Kμ(1-ρ/K))	

[Indicative Weight 20%]

Question 6

Consider the examples in the last two questions (Question 4 Database Connections and Question 5 Online Banking). This time the architect would like a more realistic modelling and they do not like the idea of any of the requests waiting in an infinite buffer. They want you to size the buffer and allow for some requests to be lost or rejected. They want to know the amount of buffer (waiting room) needed. You must move from an Erlang C model to a modified Erlang B model (M/M/K/n) with more than K places so that there is room to wait if all the connections are busy.

They would like you to compare the results with the model to the last two sets of results (from Q4 and Q5). There is still an average queue and average delay like before but this time around we also have a loss probability to consider. These should be compared to the prior questions.

[Indicative Weight 30%]