Computer Science Department

CS-350: Fundamentals of Computing Systems

Exam #1
Fall 2019

Name: ____

Login: ______________________ BU Id: ______________________

You can use the back of any page to answer the question on the front of that page

Problem #1:	*/26*
Problem #2:	*/20*
Problem #3:	*/18*
Problem #4:	*/24*
Problem #5:	*/24*
Total Grade:	*/100*

Note:
- This is a closed-book/notes exam
- Basic calculators are allowed
- Time allowed is 80 minutes
- There are 112 total points
- Problems are weighted as shown
- Sub-problems are equally weighted
- Explain your assumptions clearly

I give credit to students who can accurately assess their performance. Thus, after you finish the exam answer the following for up to five bonus points!

What do you guess your score will be? _________

Bonus = max(0 , 5 - | *GuessedGrade – ActualGrade* |)

1. Label each of the statements below with either **True** or **False**:

	Question	Answer
a.	If X is an exponentially distributed random variable, then X is also a Poisson distributed variable.	False
b.	A discrete event simulator is only capable of providing estimates for systems that can be analytically modeled, such as an M/M/1, M/M/1/K, etc.	False
c.	It is possible to apply Little's Law in any system that reaches steady state as long as no unaccounted request deaths occur in the middle of the system.	True
d.	Consider an M/M/4 system. The average response time at the system is lower than or equal to what experienced in a 4*M/M/1 system under the same conditions, i.e. same T_s, same λ.	True
e.	It is possible for a system to have a different capacity for two different types of workload.	True
f.	The throughput of an M/M/1/K system is independent from the load and it is always equal to $1/T_s$.	False
g.	It is possible to achieve a speedup of at least 2 as long as the workload is fully sequential.	False
h.	In an M/M/1/K system operating at steady with only 50% utilization, the probability of rejection is 0.	False
i.	Given an M/M/1 system for which we know the average service time to be T_s, it is possible to conclude that that the standard deviation for service time is also T_s	True
g.	In a complex system such as a network of queues, changing the routing probability for traffic inside the system **does not** affect the system's capacity.	False
k.	A process is said to be in ready state if it is currently consume the resource it needs to progress in its execution.	False
l.	The number of requests in a system at steady state is linearly proportional to the response time of the system.	True
m.	It is possible to analyze a system using the Jackson's Theorem as long as no more than two M/M/1/K system exist in the system.	False
n.	If the arrival rate of requests at a system exceed the system's capacity, it is not always possible for the system to reach steady state.	True
o.	It is possible for an accepted (i.e. placed in the queue) request to have an infinite waiting time in an M/M/1/K system.	False
p.	Consider a system where a request, after being executed, can either go out with probability p or go back into the system with probability $1-p$. On average the request will be run inside the system $1/p$ times.	True
q.	The slowdown of a request flowing through a network of queues is not affected by the utilization of the systems in the network, but only by their service time.	False
r.	Consider λ the average arrival rate of requests at an M/M/N system that is followed by an M/M/1 system. Then the average arrival rate of requests at the M/M/1 system is N*λ.	False

Note: There are 18 questions. A correct answer will get you +2 points. An incorrect answer will get you -1 points. A blank answer will get you 0 points.

2. Two weather stations communicate over a high-power wireless link. You are trying to design an application-level protocol that can allow the two stations to exchange messages. Each message can fit in a packet. You have been told that during storms, the probability that a packet transmission results in a failure is 30%. You decide to design a protocol that is able to detect a failure and perform a packet retransmission if a failure is detected. In other words, to send a single message, your protocol sends a first packet. Upon success, the protocol moves to send the next message. If failure is detected, transmission of the packet for the first message is attempted again, and so on.

(a) You want to set a maximum number of retransmissions before message transmission is aborted. What do you set this limit to, if you want a station to be able to successfully send a message during a storm with a success rate of at least 95%?

Prob of transmission success is 0.7;
N #of retransmissions.
N = 1 → message sent correctly with probability 0.7
N = 2 → message sent correctly with probability 0.7 + 0.3*0.7 = 91%
N = 3 → message sent correctly with probability 0.7 + 0.3*0.7 + (0.3^2)*0.7 = 97.3%

(b) Assuming no maximum limit on the number of retransmissions, what is the average number of transmission attempts for a message to be correctly sent?

In this case, we reason on the Geometric distribution with prob of success p = 0.8
Mean is 1/p = 1.43

(c) Considering the retransmission limit to be N = 3, what is the probability that out of 3 consecutive messages at least 2 are correctly sent?

With N = 3, reuse what computed in (a) for probability p=0.973 of a successful message send.
The compute $\binom{3}{2} p^{(2)}(1-p) + \binom{3}{3} p^{(3)} = \frac{3!}{2!(3-2)!} p^2(1-p) + p^3 = \frac{3}{2} 0.0256 + 0.9212 = 0.96$

3. A Database Management System (DBMS) is organized as a two-tier processing system. In the first tier (VAL), requests are parsed and validated (this is a single operation). Valid requests are forwarded to the second tier (PRO) for final processing. Invalid requests are rejected from the system after completing processing at the first tier. Additionally, the buffer used by the parsing/validation engine can only hold up to 8 requests. Buffer memory is released only after request validation is completed. Conversely, the final processing engine can never run out of memory. It has been observed that on average the DBMS is receiving 30 requests per second. Measurements performed on the parsing/validation stage have revealed that the operation takes on average 50 milliseconds. Additionally, 90% of the received requests are valid.

a. Show a diagram of the system, describing what model you are using to model each of the two tiers. Also clearly state what **assumptions** you are making to reason about the system.

Daisy chain of an M/M/1/K with K=8 and an M/M/1 system; output goes to the M/M/1 system with 90% prob.

Assumptions: arrivals are Poisson, service times are exponential, queue at PRO is infinite, system at steady state.

b. What is the rate at which the system rejects requests, either because they are invalid or because the validation engine's buffer is full?

Let consider the two types of rejection separately. First, compute rejections because queue is full.

\lambda = 30 req/s

\rho = 30*0.05 = 1.5

P(Sk) = ((1−1.5)×1.5^8)÷(1−1.5^(9)) = 0.34

\lambda_rfull = 30*0.34 = 10.2 req/s

\lambda_rinvalid = 30*(1-0.34)*0.1 = 1.98 req/s

\lambda_reject = 10.2 + 1.98 = 12.18 req/s

c. What is the upper bound on the average amount of time that can be spent processing an individual request at the second tier to prevent the system from "exploding"?

What makes it to the second tier is: \lambda_proc = 30 - \lambda_reject = 17.82 req/s

Alternatively, this is computed as: 30*(1-P(Sk))*0.9

We find Ts such that \lambda_proc * Ts < 1 → Ts < 0.056

d. If processing at the second stage takes 40 milliseconds, what is the average response time for all the requests that manage to go through the validation stage --- regardless of whether they are rejected because invalid?

At the first stage \lambda_accept = 30*(1-0.34) = 19.8
Then q1 = (19.8 * 0.05)/(1-19.8 * 0.05) = 99

From (c) \lambda_proc = 17.82 req/s
q2 = (17.82 * 0.04)/(1-17.82 * 0.04) = 2.48

qtot = q1 + q2 = 101.48
Tqtot = qtot/\lambda_accept = 5.13 sec

4. You have rigged an on-demand BitCoin miner in the basement of your home. People pay a subscription fee to send hash computation requests to your server. For this purpose, you have purchased a powerful 4-cores machine. When an individual core is busy processing a request, it consumes 30W of power, and only 5W when idle. It consumes 0W when powered off. After having advertised the service on Repostit, you are consistently receiving on average a new request every 5 milliseconds. Each of these requests, individually, takes on average 8 milliseconds to be processed.

a. How many cores you would need to keep powered on to be able to handle the load? Motivate your answer.

Load is: \lambda = 1/5 = 0.2 req/msec

\rho = 0.2 * 8 = 1.6 → we need at least 2 CPUs on

b. Under the assumption that you are using the minimum number of cores required to handle the load, what is the probability that a random point in time the server will be consuming only 10W?

We model the systems as an M/M/2. The probability of a single CPU to be busy is:
\rho' = 0.2*8/2 = 0.8

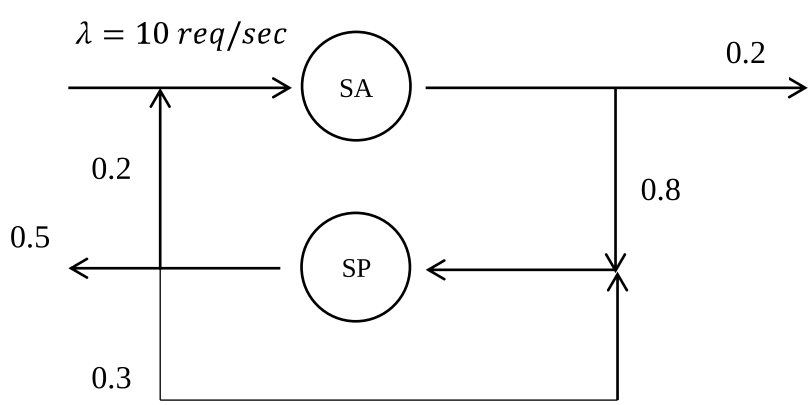
To consume only 10W, all the CPUs need to be idle.
That happens with probability (1-C)
K = $\frac{1+2*0.8}{1+2*0.8+\frac{(2*0.8)\hat{}2}{2}}$ = 0.67
C = (1-K)/(1-\rho*K) = 0.711
Finally: 1-C = 0.289

c. Eversource called with a great discount on basement coin riggers. The deal proposes a flat daily energy price for power consumption that does not exceed 85W average power consumption. Can you improve service for your users? Motivate your answer.

d. (BONUS) Quantify the improvement in terms of service you are able to provide to your users if you take any action given the discount presented above.

5. A bank transaction processing system is comprised of two servers. The first handles authentication (SA), the second (SP) performs actual processing of incoming requests. Requests that require authentication arrive at the system with an average rate of 10 requests per second. These are first handled at SA. With 20% probability, authentication fails and the request is terminated. Requests that have been successfully authenticated advance to the SP. After that, a transaction completes and leaves the system with 50% probability. Alternatively, it may require additional processing at the SP right away with 30% probability. In the remainder of the cases, the transaction not only requires further processing at the SP, but before further processing can be done, additional authentication is also required at the SA, as if it were a new request. The average time it takes the SA to perform authentication of a single transaction is X seconds, while the average service time at the SP is Y seconds.

a. Draw the queuing diagram of the above system. Specify what models you use for SA and SP.



b. At which rate requests arrive at SP?

We want to compute $\lambda_{SP} = \lambda * 0.8 + \lambda_{SP} 0.3 + \lambda_{SP} * 0.2 * 0.8$

It follows that: $\lambda_{SP} = \frac{0.8\lambda}{1-0.3-0.2*0.8} = 14.8\ \frac{req}{sec}$

c. When X = Y, what is the bottleneck of the system?

$$\lambda_{SA} = \lambda_{SP} * 0.2 + \lambda = 12.96 \frac{req}{sec}$$

This shows that SP is the bottleneck

d. What is the total number of requests in the system on average at steady state when X = 0.07 seconds and Y = 0.06 seconds?

$\rho_{SA} = \lambda_{SA} * Y = 0.9072; q_{SA} = 9.78$

$\rho_{Sp} = \lambda_{Sp} * X = 0.888; q_{SP} = 7.93$

Hence q_tot = 17.71 requests

e. What is the maximum rate of requests from the outside that the system can sustain? Once again, consider X = 0.07 seconds and Y = 0.06 seconds.

With these values of X and Y, the bottleneck is SA.

$$\lambda_{SA} = \frac{0.8\lambda}{1 - 0.3 - 0.2 * 0.8} * 0.2 + \lambda = \frac{0.16\lambda}{0.54} + \lambda = 1.29\lambda$$

We want to find λ such that $1.29\lambda * 0.07 = 1$

If follows that $\lambda = \frac{1}{1.29 * 0.07} = 11.02 \ req/sec$

[This page is intentionally blank – use as extra space]

Some Relationships:

- $$Cor(X,Y)=\frac{\frac{1}{N}\sum_{i=1}^{N}(X(i)-\overline{X})(Y(i)-\overline{Y})}{StdDev(X)*StdDev(Y)}$$
- $$AutoCor(X,d)=\frac{\frac{1}{N}\sum_{i=d+1}^{N}(X(i-d)-\overline{X})(X(i)-\overline{X})}{Var(X)}$$

Some Probability Density Functions:

- Poisson $f(x)=\frac{\lambda^x}{x!}e^{-\lambda}$
- Exponential $f(x)=\lambda e^{-\lambda x}$, $F(x)=1-e^{-\lambda x}$
- Standard Normal $f(z)=\frac{1}{\sqrt{2\pi}}e^{-0.5z^2}$

Equations for some queuing systems:

- M/G/1 system $q=\frac{\rho^2 A}{1-\rho}+\rho$ and $w=\frac{\rho^2 A}{1-\rho}$, where $A=\frac{1}{2}\left[1+\left(\frac{\sigma_{Ts}}{Ts}\right)^2\right]$
- M/D/1 system $q=\frac{\rho^2}{2(1-\rho)}+\rho$ and $w=\frac{\rho^2}{2(1-\rho)}$
- M/M/1/K system $q=\begin{cases}\frac{\rho}{(1-\rho)}-\frac{(K+1)\rho^{K+1}}{(1-\rho^{K+1})} & \text{for } \rho\neq 1\\ \frac{K}{2} & \text{for } \rho=1\end{cases}$, and

$$\Pr(\text{"Rejection"})=\Pr(S_K)=\begin{cases}\frac{(1-\rho)\rho^K}{(1-\rho^{K+1})} & \text{for } \rho\neq 1\\ \frac{1}{K+1} & \text{for } \rho=1\end{cases}$$

- M/M/N system $q=C\frac{\rho}{1-\rho}+N\rho$ and $w=C\frac{\rho}{1-\rho}$,

where $\rho=\frac{\lambda}{N\mu}=\frac{\lambda T_s}{N}$, $C=\frac{1-K}{1-\rho K}$ and $K=\frac{\sum_{i=0}^{N-1}\frac{(N\rho)^i}{i!}}{\sum_{i=0}^{N}\frac{(N\rho)^i}{i!}}$

Values of the Standard Normal Distribution (i.e. $F(z)=\int_{-\infty}^{z}\frac{1}{\sqrt{2\pi}}e^{-0.5z^2}.dz$) for z>0

z	**.00**	**.01**	**.02**	**.03**	**.04**	**.05**	**.06**	**.07**	**.08**	**.09**
0.0	.5000	.5040	.5080	.5120	.5160	.5190	.5239	.5279	.5319	.5359
0.1	.5398	.5438	.5478	.5517	.5557	.5596	.5636	.5675	.5714	.5753
0.2	.5793	.5832	.5871	.5910	.5948	.5987	.6026	.6064	.6103	.6141
0.3	.6179	.6217	.6255	.6293	.6331	.6368	.6406	.6443	.6480	.6517
0.4	.6554	.6591	.6628	.6664	.6700	.6736	.6772	.6808	.6844	6879
0.5	.6915	.6950	.6985	.7019	.7054	.7088	.7123	.7157	.7190	.7224
0.6	.7257	.7291	.7324	.7357	.7389	.7422	.7454	.7486	.7157	.7549
0.7	.7580	.7611	.7642	.7673	.7704	.7734	.7764	.7794	.7823	.7852
0.8	.7881	.7910	.7939	.7969	.7995	.8023	.8051	.8078	.8106	.8133
0.9	.8159	.8186	.8212	.8238	.8264	.8289	.8315	.8340	.8365	.8389
1.0	.8413	.8438	.8461	.8485	.8508	.8513	.8554	.8577	.8529	.8621
1.1	.8643	.8665	.8686	.8708	.8729	.8749	.8770	.8790	.8810	.8830
1.2	.8849	.8869	.8888	.8907	.8925	.8944	.8962	.8980	.8997	.9015
1.3	.9032	.9049	.9066	.9082	.9099	.9115	.9131	.9147	.9162	.9177
1.4	.9192	.9207	.9222	.9236	.9215	.9265	.9279	.9292	.9306	.9319
1.5	.9332	.9345	.9357	.9370	.9382	.9394	.9406	.9418	.9492	.9441
1.6	.9452	.9463	.9474	.9484	.9495	.9505	.9515	.9525	.9535	.9545
1.7	.9554	.9564	.9573	.9582	.9591	.9599	.9608	.9616	.9625	.9633
1.8	.9641	.9649	.9656	.9664	.9671	.9678	.9686	.9693	.9699	.9706
1.9	.9713	.9719	.9726	.9732	.9738	.9744	.9750	.9756	.9761	.9767
2.0	.9772	.9778	.9783	.9788	.9793	.9798	.9803	.9808	.9812	.9817
2.1	.9821	.9826	.9830	.9834	.9838	.9842	.9846	.9850	.9854	.9857
2.2	.9861	.9864	.9868	.9871	.9875	.9878	.9881	.9884	.9887	.9890
2.3	.9893	.9896	.9898	.9901	.9904	.9906	.9909	.9911	.9913	.9916
2.4	.9918	.9920	.9922	.9925	.9927	.9929	.9931	.9932	.9934	.9936
2.5	.9938	.9940	.9941	.9943	.9945	.9946	.9948	.9949	.9951	.9952
2.6	.9953	.9955	.9956	.9957	.9959	.9960	.9961	.9962	.9963	.9964
2.7	.9965	.9966	.9967	.9968	.9969	.9970	.9971	.9972	.9973	.9974
2.8	.9974	.9975	.9976	.9977	.9977	.9978	.9979	.9979	.9980	.9981
2.9	.9981	.9982	.9982	.9983	.9984	.9984	.9985	.9985	.9986	.9986
3.0	.9987	.9987	.9987	.9988	.9988	.9989	.9989	.9989	.9990	.9990
3.1	.9990	.9991	.9991	.9991	.9992	.9992	.9992	.9992	.9993	.9993
3.2	.9993	.9993	.9994	.9994	.9994	.9994	.9994	.9995	.9995	.9995
3.3	.9995	.9995	.9995	.9996	.9996	.9996	.9996	.9996	.9996	.9997
3.4	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9998