Computer System Fundamentals HW #4

Quan Zhou

Feb 18th, 2016

Problem 1

(a) Since the streaming media buffer system has:

. Arrival of the packets to the application follows Poisson process;
. The processing time for each packet is an exponentially distributed variable;
. Only one server processing;
. The buffer can hold up to K packets, or the system size is finite;

we can model the above system as an M/M/1/K system.

(b) $\rho = \lambda T_s = (50 \text{ packets per second})(20 \text{ msec}) = (50 \text{ packets per second})(0.02 \text{ sec}) = 1 \text{ packet}$
Since $\rho = 1$, $q = \frac{K}{2} = 1.5$

(c) $\Pr(S_K) = \frac{1}{K+1} = \frac{1}{4}$
$\Pr(\bar{S}_K) = 1 - \frac{1}{K+1} = \frac{3}{4}$
so out of 2000 packets, there are 500 blips and 1500 plops.

(d)

$$T_q = \frac{q}{\lambda\prime} = \frac{q}{\lambda(1 - \Pr(S_k))} = \frac{1.5}{0.05 \text{ packets per msec}(1 - \frac{1}{4})} = 40 \text{ msec}$$

(e) If K = 10:

$$q = \frac{K}{2} = 5$$

$$\Pr(S_K) = \frac{1}{K+1} = \frac{1}{11}$$

$$T_q\prime = \frac{q}{\lambda\prime} = \frac{q}{\lambda(1 - \Pr(S_k\prime))} = \frac{5}{0.05 \text{ packets per msec}(1 - \frac{1}{11})} = 110 \text{ msec}$$

(f) As K increases, the buffer system size increases so that more packets are likely to be plopped in the played out and waited in the queue. So the mean response time/delay time for the packet from arrival to being serviced increases.
In general, the mean delay for packets that are played out to increase.

(g) `"Increasing the size of network buffers (i.e., K) has its advantages and its disadvantages".` clearly the advantage of larger size of network buffer is less blips and more plops; but the disadvantage is the mean delay increases and resulting in an extended buffering process.

(h) For "Movie on Demand" streaming application like Netflix, larger buffer size (or larger K) is preferred because we do not want to have dropped the video data packets (or blips) making the video not watchable. It is OK for delay time to be increased as long as the quality of video is not compromised.
For teleconferencing application like Skype, smaller buffer size would be more preferable because even though blips could affect the voice data being transferred the audiences could still catch the meaning thus making the audio understandable. In cases like this having longer delay is bad because it delays the communication. In general, it is better to hear something immediately than nothing for a while.

Problem 2

(a) Base on the results from sim1 log.txt, we imported the data into Excel and analyzed column Tq, and q (which is calculated from λT_q). We found the mean μ and standard deviation s to be (0.039, 0.032) sec and q (3.13 , 1.59) requests.
Assuming the data being normal and use the formula below to find confidence interval:

$$\left[\bar{X} - z\frac{\sigma}{\sqrt{n}}, \bar{X} + z\frac{\sigma}{\sqrt{n}}\right] \tag{1}$$

so the confidence level for q is $[0.008, 6.24]$ and T_q $[-0.023, 0.102]$
Compared to :

$$\rho = \lambda T s = 50 * 0.02 = 1$$
$$q = \frac{K}{2} = 3/2 = 1.5$$
$$w = q - \rho = 1.5 - 1 = 0.5$$
$$T_q = \frac{q}{\lambda} = \frac{1.5}{50} = 0.03$$
$$T_w = \frac{w}{\lambda} = \frac{0.5}{50} = 0.01$$
$$\Pr(\text{"Rejection"}) = \frac{1}{K+1} = 0.25$$

We found 1.5 falls in the 95% confidence interval $[0.008, 6.24]$ for q and 0.03 in $[-0.023, 0.102]$ for T_q.

(b) Base on the results from sim2 log.txt, we imported the data into Excel and analyzed column Tq, and q (which is calculated from λT_q). We found the mean μ and standard deviation s to be (0.028, 0.024) sec and q (1.406 , 1.206) requests.
Using equation from part (a), we calculate the 95% confidence level for q is $[0, 3.77]$ and T_q $[-0.019, 0.072]$
Compared to :

$$\rho = \lambda T s = 50 * 0.015 = 0.75$$
$$q = \frac{\rho}{1-\rho} - \frac{(K+1)\rho^{K+1}}{1-\rho^{K+1}} = 3 - \frac{40.75^4}{1-0.75^4} = 1.15$$
$$w = q - \rho = 1.15 - 0.75 = 0.4$$
$$T_q = \frac{q}{\lambda} = \frac{1.15}{50} = 0.023$$
$$T_w = \frac{w}{\lambda} = \frac{0.4}{50} = 0.008$$

$$\Pr(\text{"Rejection"}) = \frac{(1-\rho)\rho^K}{1-\rho^{K+1}} = \frac{(0.25)0.75^3}{1-0.75^4} = 0.1542$$

We found 1.15 falls in the 95% confidence interval $[0, 3.77]$ for q and 0.023 in $[-0.019, 0.072]$ for T_q.

(c) Base on the results from sim3 log.txt, we imported the data into Excel and analyzed column Tq, and q (which is calculated from λT_q). We found the mean μ and standard deviation s to be (0.030, 0.024) sec and q (1.963 , 1.589) requests.
so the confidence level for q is $[0, 5.078]$ and T_q $[-0.018, 0.078]$
Compared to :

$$\rho = \lambda Ts = 65 * 0.015 = 0.975$$

$$q = \frac{\rho}{1-\rho} - \frac{(K+1)\rho^{K+1}}{1-\rho^{K+1}} = 39 - \frac{40.975^4}{1-0.975^4} = 1.468$$

$$w = q - \rho = 1.468 - 0.975 = 0.493$$

$$T_q = \frac{q}{\lambda} = \frac{1.468}{65} = 0.0226$$

$$T_w = \frac{w}{\lambda} = \frac{0.493}{65} = 0.0076$$

$$\Pr(\text{"Rejection"}) = \frac{(1-\rho)\rho^K}{1-\rho^{K+1}} = \frac{(0.025)0.975^3}{1-0.975^4} = 0.2346$$

We found 1.468 falls in the 95% confidence interval $[0, 5.078]$ for q and 0.0226 in $[-0.018, 0.078]$ for T_q.

(d) Base on the results from sim4 log.txt, we imported the data into Excel and analyzed column Tq, and q (which is calculated from λT_q). We found the mean μ and standard deviation s to be (0.024, 0.009) sec and q (1.561 , 0.610) requests.
so the confidence level for q is $[0.364, 2.757]$ and T_q $[0.006, 0.042]$
Compared to the confidence level for q is $[0.008, 6.24]$ and T_q $[-0.023, 0.102]$ found in part (a):
Because of the huge variation from the run in part a, it is unclear if the rejection probability varied because of the change in the distribution of service time. But intuitively, the rejection probability given this M/D/1/K system should be LESS: Since the utilization for part (d) is less than that of part (a), it is less likely for the system in (d) to reject a process because it has higher change of idle time.

(e) Base on the results from sim5 log.txt, we imported the data into Excel and analyzed column Tq, and q (which is calculated from λT_q). We found the mean μ and standard deviation s to be (0.028, 0.010) sec and q (1.806 , 0.642) requests.
so the confidence level for q is $[0.547, 3.065]$ and T_q $[0.008, 0.047]$
Compared to the confidence level for q $[0, 3.77]$ and T_q $[-0.019, 0.072]$ from part (b):
Because of the huge variation from the run in part a, it is unclear if the rejection probability varied because of the change in the distribution of service time. But intuitively, the rejection probability given this M/D/1/K system at arrival rate being 65 should be MORE: Given all things equal, it is more likely for the system to reject an incoming process because it has higher rate

`Results from M/M/1 system in Assignment 3:`

(a) part b (HW3) versus part b (HW4)
Given the same settings: $\lambda = 50$, $T_s = 0.015$, and Simulation time is 100, M/M/1 and M/M/1/K behaves somewhat similar. As we calculated the ρ to be 0.75, which resulted $q = 3$ for M/M/1 system. In a way it behaves like the M/M/1/K system with the queue size of 3. and turns out the mean value of q and T_q are about the same.

(b) part c (HW3) versus part c (HW4)
Though the settings ($\lambda = 65$, $T_s = 0.015$, and Simulation time is 100), M/M/1 system without buffer limit was able to surpass that of M/M/1/K with $K = 3$. In M/M/1, mean value of q is about 40 empirically (analytically too be 39) where as in M/M/1/K (K = 3) is only 1.15. These makes sense as it shows how much the queue size limits the performance of the server even though utilization is high.

NOTE:\\All the log files were printed from the console, copied in these txt files and then imported in excel. Due to time constraints, I was not able to generate auto-log and auto-import codes for M/M/1/K and M/D/1/K simulations.

Problem 3

(a) Recall the diagram from HW3:

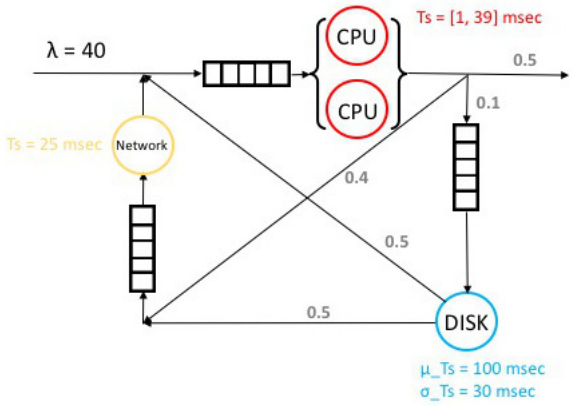
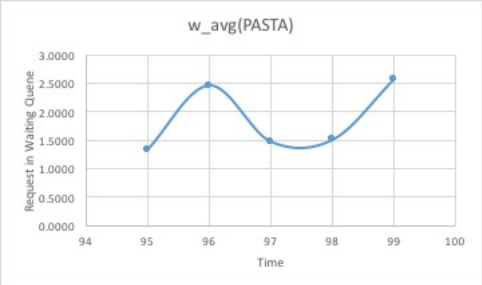


Figure 1. Queueing network with modification

We have modified the computation based on problem 3 and ran the java code (courtesy to Auwong) and obtained a log of the simulation:
Final Results of simulation:
T_w: 0.5424771197409428
T_q: 0.563099166123886
w_{cpu}: 1.5454545454545454
q_{cpu}: 3.164869029275809
w_{disk}: 3.6512583461736003
q_{disk}: 4.49306625577812
w_{net}: 12.676938880328711
q_{net}: 13.622496147919877

Due to large quantity of data, was only able to print the last few seconds (assume the unit of time is second) out of 100 (See Figure 2).



(b) The system may reach the steady state long before the last few seconds, but i am confident that this CPU/DISK/network system has reached steady state by the end of 100 seconds.

(c) The Z-scores for 95% and 98% are 1.96 and 2.32 respectively. I calculated the confidence interval in the code (see LogEvent.java) and obtained the following:
`95% Confidence Interval:`
For q: [3.1659-5.5996, 3.1659+5.5996]
For Tq: [0.5631-1.4925, 0.5631+1.4925]

`98% Confidence Interval:`
For q: [3.1659-6.6281, 3.1659+6.6281]
For Tq: [0.5631-1.7667, 0.5631+1.7667]

The simulation somehow has large variation for the values above.

(d) From HW3, I calculated the following metrics below:

$$x = 0.5x + 40 \tag{2}$$

so $x = 80$ processes per second $= 0.08$ processes per msec.
Now we can find:

$$\rho_{CPU} = \frac{1}{2}\lambda_{CPU}T_s(CPU) = \frac{1}{2}(0.08)(20) = 0.8(versus\frac{1}{2} * (3.1649 - 1.5454) = 0.8097)$$
$$\rho_{DISK} = \lambda_{DISK}T_s(DISK) = (0.1)(0.08)(100) = 0.8(versus4.4931 - 3.6513 = 0.8418)$$
$$\rho_{Network} = \lambda_{Network}T_s(Network) = [(0.4)(0.08) + (0.1)(0.08)(0.5)](25) = 0.9(versus13.6225 - 12.6769 = 0.9456)$$
$$q_{CPU} = \frac{\rho_{CPU}}{1 - \rho_{CPU}} = 4(versus3.16)$$
$$q_{DISK} = 4(versus4.49)$$
$$q_{Network} = 9(versus13.62)$$

So simulation results make sense and are very close to the analytical results
`NOTE: parentheses are the simulation results\\`

Problem 4

(a) When I modified λ to 1 request per second, i obtained the following results:
98th Confidence level E = 0.036675890831936536
98th Confidence level E = 0.0288449709524687
Final Results of simulation:

T_w: 0.044206714195278024
T_q: 0.05780577574711234
w_{cpu}: 0.0
q_{cpu}: 0.018867924528301886
w_{disk}: 0.0
q_{disk}: 0.018867924528301886
w_{net}: 0.0
q_{net}: 0.018867924528301886
95 % Confidence level of q: [0.01887-0.0310, 0.01887+0.0310]
95 % Confidence level of Tq: [0.0578-0.0244, 0.0578+0.0244]
Therefore, the slowdown of the system ($\lambda = 40$) is :

$$\text{Slowdown} = \frac{T_q}{T_s} = \frac{T_q(\lambda = 40}{T_q(\lambda \to 1} = \frac{0.5631}{0.0578} = 9.74$$

(b) From the calculations in HW3, it is known that the network is the bottleneck for it would hit 100% utilization first. Let the rate going through CPU be x processes per second. I have the following flowrate balance for Network server:

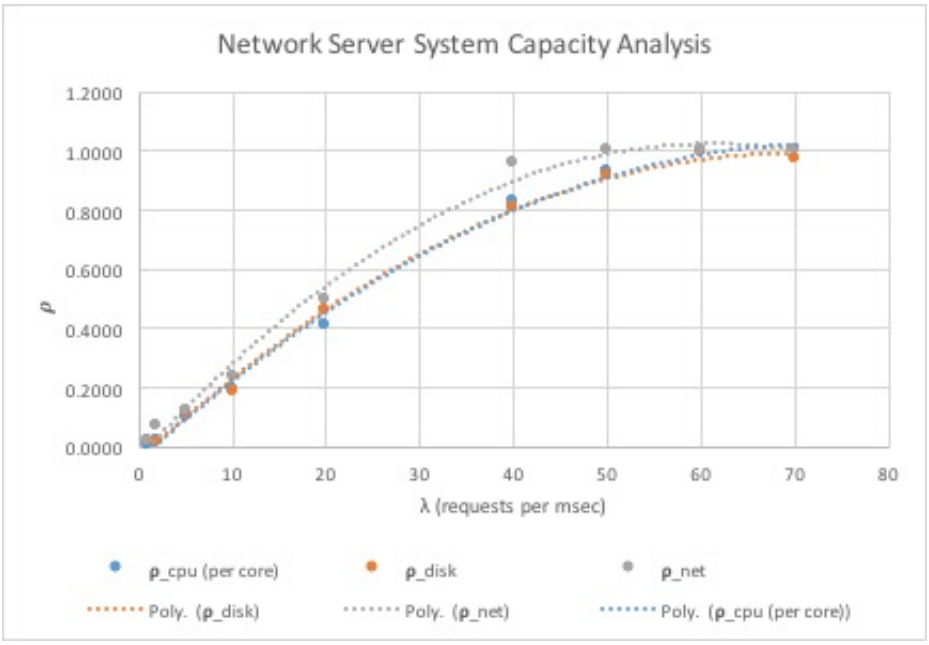
$$[0.4(x) + 0.1(x)(0.5)]25 = 1$$

so x is found to be $\frac{4}{45}$ or 0.088 requests per msec, which is our upper limit of λ.
I collected the simulation results given λ in the range from 1 to 70 requests/msec (Table 1) and then plotted the utilization of all three serving units in this system.

Table 1. Simulation runs at different Incoming Request Rate

λ	Tw	Tq	w_cpu	q_cpu	ρ_cpu (per core)	w_disk	q_disk	ρ_disk	w_net	q_net	ρ_net
1	0.0442	0.0578	0.0000	0.0189	**0.0094**	0.0000	0.0189	0.0189	0.0000	0.0189	0.0189
2	0.0591	0.0823	0.0000	0.0455	**0.0227**	0.0000	0.0114	0.0114	0.0000	0.0682	0.0682
5	0.0812	0.1062	0.0000	0.2009	**0.1005**	0.0000	0.1142	0.1142	0.0091	0.1324	0.1233
10	0.0734	0.0934	0.0020	0.3892	**0.1936**	0.0220	0.2096	0.1876	0.0379	0.2735	0.2355
20	0.0982	0.1178	0.1124	0.9345	**0.4110**	0.2825	0.7429	0.4604	0.2590	0.7595	0.5005
40	0.4306	0.4509	1.6693	3.3218	**0.8263**	1.7225	2.5269	0.8043	11.2247	12.1868	0.9621
50	4.5654	4.5843	4.9108	6.7706	**0.9299**	5.9713	6.8849	0.9136	243.7236	244.7236	1.0000
60	9.0161	9.0322	315.4419	317.4419	**1.0000**	17.8719	18.8696	0.9977	480.8873	481.8873	1.0000
70	12.7800	12.7940	1165.6925	1167.6925	**1.0000**	4.5831	5.5534	0.9703	344.3075	345.3075	1.0000

The simulation agrees with the analytical results in which the network server hits 100% utilization first but at $\lambda = 50$.


Network Server System Capacity Analysis
1.2000
1.0000
0.8000
0.6000
0.4000
0.2000
0.0000
ρ
0
10
20
30
40
50
60
70
80
λ (requests per msec)
ρ_cpu (per core)
ρ_disk
ρ_net
Poly. (ρ_disk)
Poly. (ρ_net)
Poly. (ρ_cpu (per core))