1) Suppose an interactive system is supporting 100 users with 15 second think times and a system throughput of 5 interactions/second.

a. What is the response time of the system?

b. Suppose that the service demands of the workload evolve over time so that system throughput drops to 50% of its former value (i.e., to 2.5 interactions/second). Assuming that there still are 100 users with 15 second think times, what would their response time be?

c. How do you account for the fact that response time in (b) is more than twice as large as that in (a)?

a) Response time law : R = (N/X) –Z

= 100/5 – 15 = 5s

b) R = 100/2.5 – 15 = 25s

c) Response time is inversely proportional to throughput. If response time is low, number of responses per second will be high

2) You're going to develop a bounded model for the proxy and web server system. You should follow Algorithm 5.1 in the book "Quantitative System Performance" (PDF on the Moodle).

1. Write down the visit count for the Proxy, Router, Web server A and B; these will be as a function of the probability of a hit in the proxy. Write this down in your PDF writeup. You need to properly account for the fact that you either use either web server A or B, but not both, averaged.

   If N is number of messages transferred

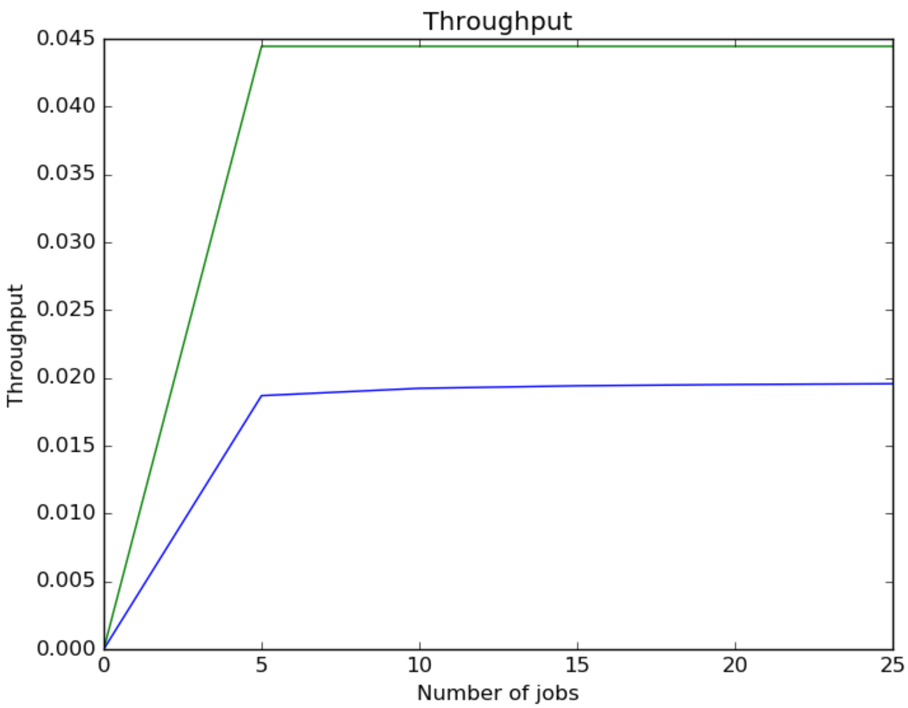
   VC(proxy) = N
   VC(router) = 0.3*2 = 0.6N
   VC(OriginA)= 0.3*0.5 = 0.15N
   VC(OriginB)= 0.3*0.5 = 0.15N

2. Write down the Demand (D_i) for each service center - this is based on the visits V_i and the service time S_i. The "delay center" for the transmission time should be included in the total demand D, but is not eligible for the maximum delay center D_{max} because there is no queuing for the transmission time, so it's not the bottleneck.

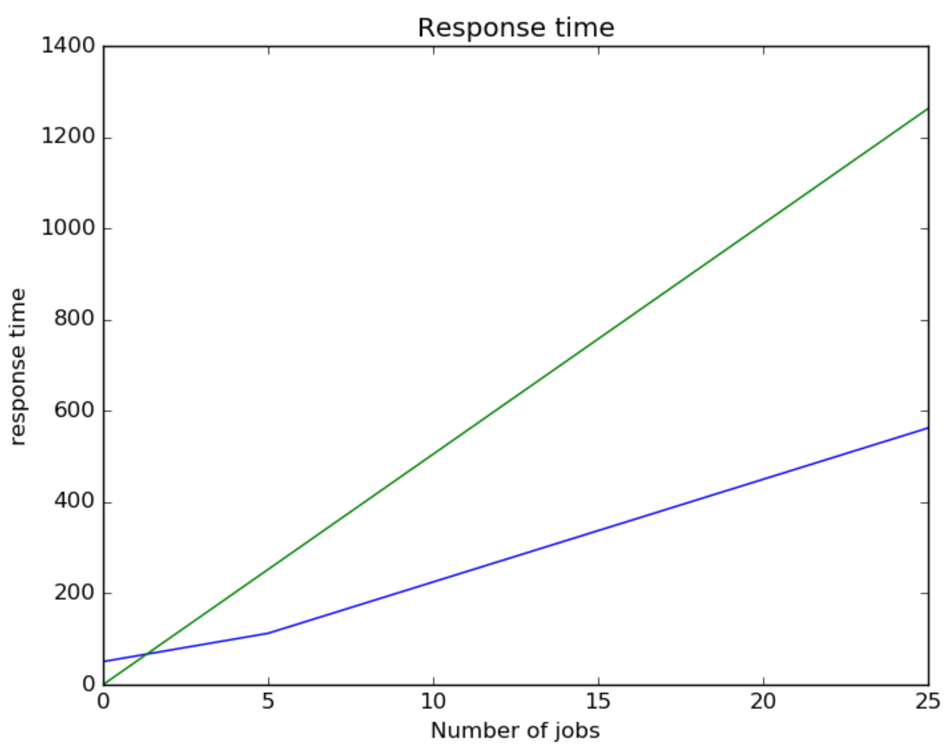
   Demand Proxy = 1*10s = 10
   Demand router = 0.6*5s = 3
   Demand OriginA= 0.15*150s= 22. 5
   Demand OriginB= 0.15*100s= 15

3. Examine case study 5.3.1 for the process of calculating the job bounds, and calculate the upper bounds on throughput (X) and the lower bounds on response time (R). Write the bounds down and also graph those bounds (two lines for each).

Throughput

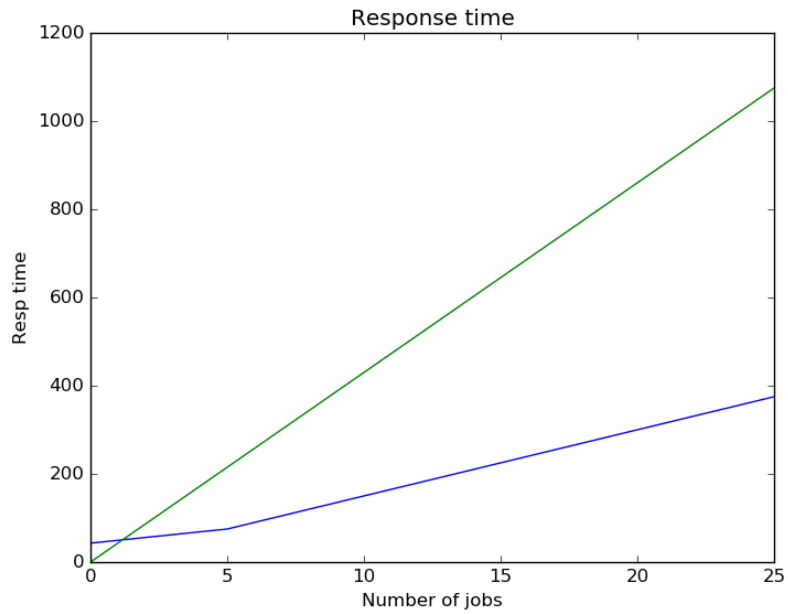


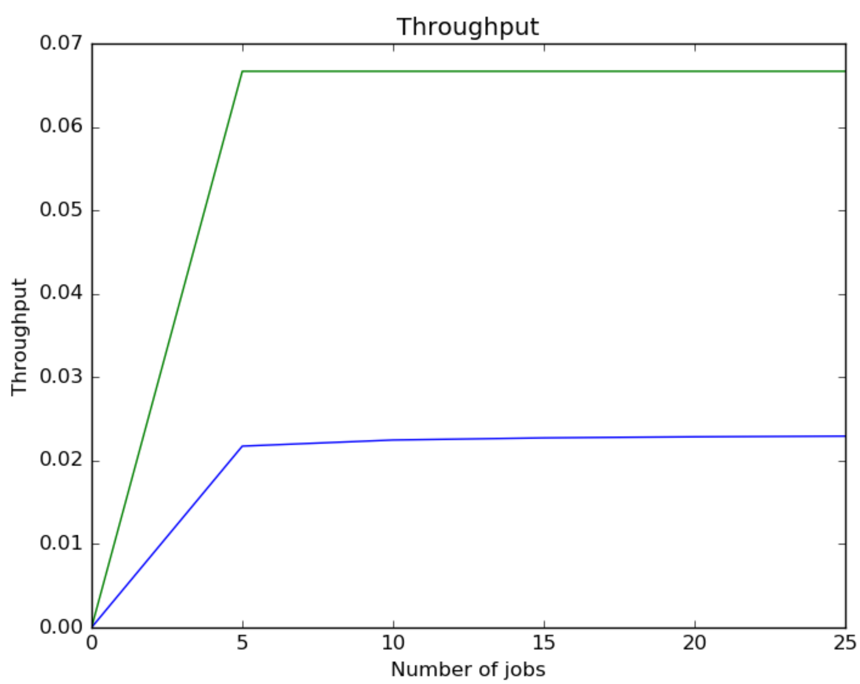
Response Time



```
import matplotlib.pyplot as plt
Dmax=22.5
D=50.5
Z=15

def AsymptoticlowboundThroughput(N):
	low = N/(N*D+Z)
	return low

def AsymptoticlowboundResponse(N):
	return max(N*Dmax,D)

def AsymptoticupboundThroughput(N):
#1/Dmax
	up = N/(D+Z)
	return min(1/Dmax,up)

def AsymptoticupboundResponse(N):
	return N*D


rlow=[]; rup=[]; tlow=[]; tup=[]
for i in range(0,30,5):
	rlow.append(AsymptoticlowboundResponse(i))
	rup.append(AsymptoticupboundResponse(i))
	tlow.append(AsymptoticlowboundThroughput(i))
	tup.append(AsymptoticupboundThroughput(i))
plt.plot(range(0,30,5),rlow)
plt.plot(range(0,30,5),rup)
#plt.plot(range(0,30,5),tlow)
#plt.plot(range(0,30,5),tup)
plt.show()
```

4. Modify the model in a method similar to 5.3.3 of the QSP text. In your case, assume that the two origin servers are of equal performance. Write down the bounds equations and graph them again, overlaying them with the original bounds from part 3.





5. Now, determine the lower bounds on throughput (X) and the upper bounds on response time (R) for the original system using the balanced job bounds equations summarized in Table 5.2. Produce a single graph with the upper and lower bounds of X and R.

3) You're going to develop a Mean Value Analysis model for the proxy and web server system, ideally as an Excel spreadsheet. You will be using the "Exact Solution Technique" described in section 6.4.2 of the QSP text. You need to do the following:

1. You will use the V_i and D_i from the prior problem.
2. Write down the equations for the Response for the individual components for N customers in the system, using the appropriate service times (again, in the PDF document) and the Queue length for N-1 customers in the system

   Rproxy = Dproxy * (1+Qproxy(N-1))
   Rrouter = Drouter * (1+Qrouter(N-1))
   Ra = Da * (1+Qa(N1))
   Rb = Db * (1+Qb(N1))

   Qproxy(N) = X(N) * Rproxy(N)
   Qrouter(N) = X(N) * Rrouter(N)
   Qa(N) = X(N) * Ra(N)
   Qb(N) = X(N) * Rb(N)

   Demand Proxy = 1*10s = 10
   Demand router = 0.6*5s = 3
   Demand OriginA= 0.15*150s= 22. 5
   Demand OriginB= 0.15*100s= 15

3. Write down the equation for the System Response, which combines the visit ratios and the individual response times

   Responetime=N/(X-Z)

4. Write down the equation for the Throughput.

   X=N/(z+Sum(R))

5. Write down the equations for the Q (number in queue) for N customers in the system

   Q=X*R

6. Now, wrap all that in into a lovely Excel spreadsheet and drag out the computation until at least N=30; cut and paste or do whatever it is that gets it into your single PDF document that is easy to grade. The example in Table 6.2 of the QSP book shows the structure to follow (although that example is "sideways", or use the PPT example from class.

N	Rproxy	Rrouter	Ra	Rb	Qproxy	Qrouter	Qa	Qb	X
0					0	0	0	0	
1	10	3	22.5	15	0.152672	0.045802	0.343511	0.229008	0.015267
2	11.52672	3.137405	30.22901	18.43511	0.294318	0.080109	0.771855	0.470714	0.025534
3	12.94318	3.240327	39.86673	22.06072	0.417024	0.104402	1.284491	0.710788	0.03222
4	14.17024	3.313206	51.40105	25.66182	0.517416	0.120979	1.87687	0.937022	0.036514
5	15.17416	3.362938	64.72957	29.05533	0.595897	0.132064	2.541964	1.141018	0.039271
6	15.95897	3.396193	79.69418	32.11526	0.655109	0.139412	3.271415	1.318319	0.04105
7	16.55109	3.418237	96.10684	34.77479	0.698565	0.144272	4.05634	1.467724	0.042207
8	16.98565	3.432816	113.7677	37.01587	0.729773	0.147488	4.887924	1.590353	0.042964
9	17.29773	3.442464	132.4783	38.8553	0.751807	0.149619	5.757873	1.688759	0.043463
10	17.51807	3.448857	152.0522	40.33138	0.767157	0.151033	6.658719	1.766205	0.043792
11	17.67157	3.4531	172.3212	41.49308	0.777739	0.151974	7.583984	1.826142	0.044011
12	17.77739	3.455921	193.1396	42.39213	0.784975	0.152599	8.528232	1.871857	0.044156
13	17.84975	3.457797	214.3852	43.07786	0.789891	0.153015	9.48702	1.906291	0.044252
14	17.89891	3.459045	235.958	43.59436	0.793215	0.153292	10.4568	1.931944	0.044316
15	17.93215	3.459877	257.7781	43.97916	0.795454	0.153477	11.43481	1.950877	0.044359
16	17.95454	3.460431	279.7831	44.26315	0.796959	0.1536	12.41889	1.964734	0.044388
17	17.96959	3.4608	301.9251	44.47101	0.797967	0.153682	13.40745	1.974803	0.044407
18	17.97967	3.461047	324.1676	44.62205	0.798642	0.153737	14.39926	1.982075	0.044419
19	17.98642	3.461211	346.4833	44.73112	0.799094	0.153773	15.39342	1.987296	0.044428
20	17.99094	3.46132	368.852	44.80945	0.799395	0.153798	16.38928	1.991028	0.044433
21	17.99395	3.461393	391.2588	44.86542	0.799596	0.153814	17.38635	1.993683	0.044437
22	17.99596	3.461441	413.6929	44.90524	0.799731	0.153825	18.38429	1.995564	0.044439
23	17.99731	3.461474	436.1465	44.93347	0.799821	0.153832	19.38284	1.996893	0.044441
24	17.99821	3.461495	458.6138	44.9534	0.79988	0.153837	20.38182	1.997829	0.044442
25	17.9988	3.46151	481.091	44.96744	0.79992	0.15384	21.38111	1.998486	0.044443
26	17.9992	3.461519	503.575	44.9773	0.799947	0.153842	22.38061	1.998947	0.044443
27	17.99947	3.461526	526.0638	44.9842	0.799965	0.153843	23.38027	1.999268	0.044444
28	17.99965	3.46153	548.556	44.98902	0.799976	0.153844	24.38003	1.999492	0.044444
29	17.99976	3.461533	571.0506	44.99239	0.799984	0.153845	25.37986	1.999648	0.044444
30	17.99984	3.461535	593.5469	44.99473	0.799989	0.153845	26.37974	1.999757	0.044444

7. Now, repeat the analysis for a system where the origin servers have balanced service times. If you structure your spreadsheet correctly, this should be just changing a single cell that defines the service time of the web servers.

N	Rproxy	Rrouter	Ra	Rb	Qproxy	Qrouter	Qa	Qb	X
0					0	0	0	0	
1	10	3	15	15	0.172414	0.051724	0.258621	0.258621	0.017241
2	11.72414	3.155172	28.31897	18.87931	0.304217	0.08187	0.734817	0.489878	0.025948
3	13.04217	3.24561	39.03339	22.34817	0.422216	0.105071	1.263634	0.723481	0.032373

4	14.22216	3.315212	50.93177	25.85222	0.52038	0.121302	1.863562	0.945916	0.036589
5	15.2038	3.363905	64.43014	29.18875	0.597697	0.132243	2.532898	1.147477	0.039312
6	15.97697	3.396729	79.49021	32.21216	0.656246	0.139519	3.26502	1.323098	0.041074
7	16.56246	3.418557	95.96295	34.84647	0.6993	0.144338	4.051745	1.471287	0.042222
8	16.993	3.433015	113.6643	37.0693	0.730255	0.14753	4.884595	1.593012	0.042974
9	17.30255	3.44259	132.4034	38.89518	0.752126	0.149646	5.755454	1.690738	0.043469
10	17.52126	3.448939	151.9977	40.36107	0.767369	0.151051	6.656961	1.767672	0.043796
11	17.67369	3.453154	172.2816	41.51507	0.77788	0.151985	7.58271	1.827222	0.044013
12	17.7788	3.455956	193.111	42.40833	0.785069	0.152607	8.527312	1.872649	0.044158
13	17.85069	3.45782	214.3645	43.08973	0.789954	0.15302	9.486358	1.906867	0.044253
14	17.89954	3.459061	235.9431	43.60301	0.793257	0.153296	10.45633	1.932362	0.044317
15	17.93257	3.459887	257.7674	43.98543	0.795482	0.153479	11.43447	1.951177	0.04436
16	17.95482	3.460438	279.7755	44.26766	0.796977	0.153602	12.41865	1.964949	0.044388
17	17.96977	3.460805	301.9197	44.47424	0.79798	0.153683	13.40728	1.974957	0.044407
18	17.9798	3.46105	324.1638	44.62436	0.79865	0.153738	14.39914	1.982184	0.044419
19	17.9865	3.461213	346.4806	44.73276	0.799099	0.153774	15.39334	1.987374	0.044428
20	17.99099	3.461321	368.8501	44.81061	0.799399	0.153798	16.38922	1.991083	0.044433
21	17.99399	3.461394	391.2575	44.86624	0.799599	0.153814	17.38631	1.993721	0.044437
22	17.99599	3.461442	413.692	44.90582	0.799733	0.153825	18.38426	1.995592	0.044439
23	17.99733	3.461474	436.1458	44.93387	0.799822	0.153832	19.38282	1.996912	0.044441
24	17.99822	3.461496	458.6134	44.95369	0.799881	0.153837	20.38181	1.997843	0.044442
25	17.99881	3.46151	481.0906	44.96764	0.799921	0.15384	21.3811	1.998496	0.044443
26	17.99921	3.461519	503.5747	44.97743	0.799947	0.153842	22.38061	1.998953	0.044443
27	17.99947	3.461526	526.0636	44.9843	0.799965	0.153843	23.38026	1.999273	0.044444
28	17.99965	3.46153	548.5559	44.98909	0.799976	0.153844	24.38002	1.999496	0.044444
29	17.99976	3.461533	571.0505	44.99243	0.799984	0.153845	25.37986	1.999651	0.044444
30	17.99984	3.461535	593.5468	44.99476	0.79999	0.153845	26.37974	1.999758	0.044444

8. Compare the results from the simulation model, MVA model and the balanced job bounds. Do the bounds correctly bound the MVA model? How different are the MVA results than the simulation?

   Bounds are proper bound MVA model.

4) Approximate mean value analysis is useful when you want to compute the performance for a system of N jobs, but aren't interested in systems with 0, 1, 2, ..., N-1 jobs. The QSP text shows one way to conduct approximate MVA (AMVA) -- you "spread" the N jobs through the different queue centers (Q = (N/K)) and then iterate on pushing those jobs to their resulting locations until steady state is reached. In this system, the iterations in the model represent iterations of a single model, not the values for different numbers of jobs in the system.

1. The equations are similar to those for part 2 of the homework, and you should start with that spreadsheet. Modify the calculations of the Q values following Algorithm 6.3 of the QSP text.
2. Continue the model until the queue lengths stabelize, but for no more than 30 iterations (or rows in Excel).

N	Rproxy	Rrouter	Ra	Rb	Qproxy	Qrouter	Qa	Qb
0					0	0	0	0
					0.17241	0.05172	0.25862	0.25862
1	10	3	15	15	4	4	1	1
	11.7241	3.15517	28.3189	18.8793	0.15210	0.04093	0.36740	0.24493
2	4	2	7	1	8	5	9	9
	11.5210	3.12280	30.7666	18.6740	0.43704	0.11846	1.16710	0.70838
3	8	5	9	9	1	1	5	3
	14.3704	3.35538	48.7598	25.6257	0.53665	0.12530	1.82090	0.95697
4	1	2	5	5	3	4	3	6
	15.3665	3.37591	63.4703	29.3546	0.60704	0.13336	2.50737	1.15964
5	3	3	1	3	9	4	2	5
	16.0704	3.40009	78.9158	32.3946	0.66142		3.24798	1.33328
6	9	3	7	7	3	0.13994	7	6
	16.6142	3.41981	95.5797		0.70223	0.14454	4.03988	1.47932
7	3	9	1	34.9993	7	6	7	2
	17.0223	3.43363	113.397	37.1898	0.73197	0.14764	4.87617	1.59919
8	7	8	5	3	5	9	5	1
	17.3197	3.44294	132.213	38.9878	0.75316	0.14971	5.74941	1.69541
9	5	7	9	6	2	9	8	5
	17.5316	3.44915	151.861	40.4312	0.76800	0.15109	6.65261	1.77117
10	2	7	9	3	8	7	8	2
	17.6800	3.45329	172.183	41.5675	0.77828	0.15201	7.57958	1.82981
11	8	2	9	7	2	5	3	6
	17.7828	3.45604	193.040	42.4472	0.78532	0.15262	8.52506	1.87455
12	2	5	6	4	5	6	3	6
	17.8532	3.45787	214.313	43.1183	0.79011	0.15303	9.48474	1.90825
13	5	8	9	4	9	3	4	9
	17.9011	3.45909	235.906	43.6238	0.79336	0.15330	10.4551	1.93337
14	9	9	7	8	5	4	7	1
	17.9336	3.45991	257.741	44.0005	0.79555	0.15348	11.4336	1.95190
15	5	2	4	6	3	5	4	5
	17.9555	3.46045		44.2785	0.79702	0.15360	12.4180	1.96547
16	3	5	279.757	8	4	5	7	1
	17.9702	3.46081	301.906	44.4820		0.15368	13.4068	
17	4	6	5	7	0.79801	6	6	1.97533
		3.46105	324.154	44.6299	0.79867	0.15373	14.3988	1.98244
18	17.9801	7	4	5	1	9	4	9
	17.9867	3.46121		44.7367	0.79911	0.15377	15.3931	1.98756
19	1	7	346.474	4	3	5	3	2

20	17.99113	3.461324	368.8455	44.81343	0.799408	0.153799	16.38908	1.991215
21	17.99408	3.461396	391.2542	44.86823	0.799605	0.153814	17.38621	1.993815
22	17.99605	3.461443	413.6897	44.90722	0.799737	0.153825	18.38419	1.995657
23	17.99737	3.461475	436.1442	44.93486	0.799824	0.153832	19.38277	1.996958
24	17.99824	3.461496	458.6123	44.95438	0.799883	0.153837	20.38177	1.997875
25	17.99883	3.46151	481.0899	44.96812	0.799922	0.15384	21.38108	1.998518
26	17.99922	3.46152	503.5742	44.97777	0.799948	0.153842	22.38059	1.998969
27	17.99948	3.461526	526.0633	44.98453	0.799965	0.153843	23.38025	1.999283
28	17.99965	3.46153	548.5556	44.98925	0.799977	0.153844	24.38002	1.999503
29	17.99977	3.461533	571.0504	44.99255	0.799985	0.153845	25.37985	1.999656
30	17.99985	3.461535	593.5467	44.99484	0.79999	0.153845	26.37974	1.999762