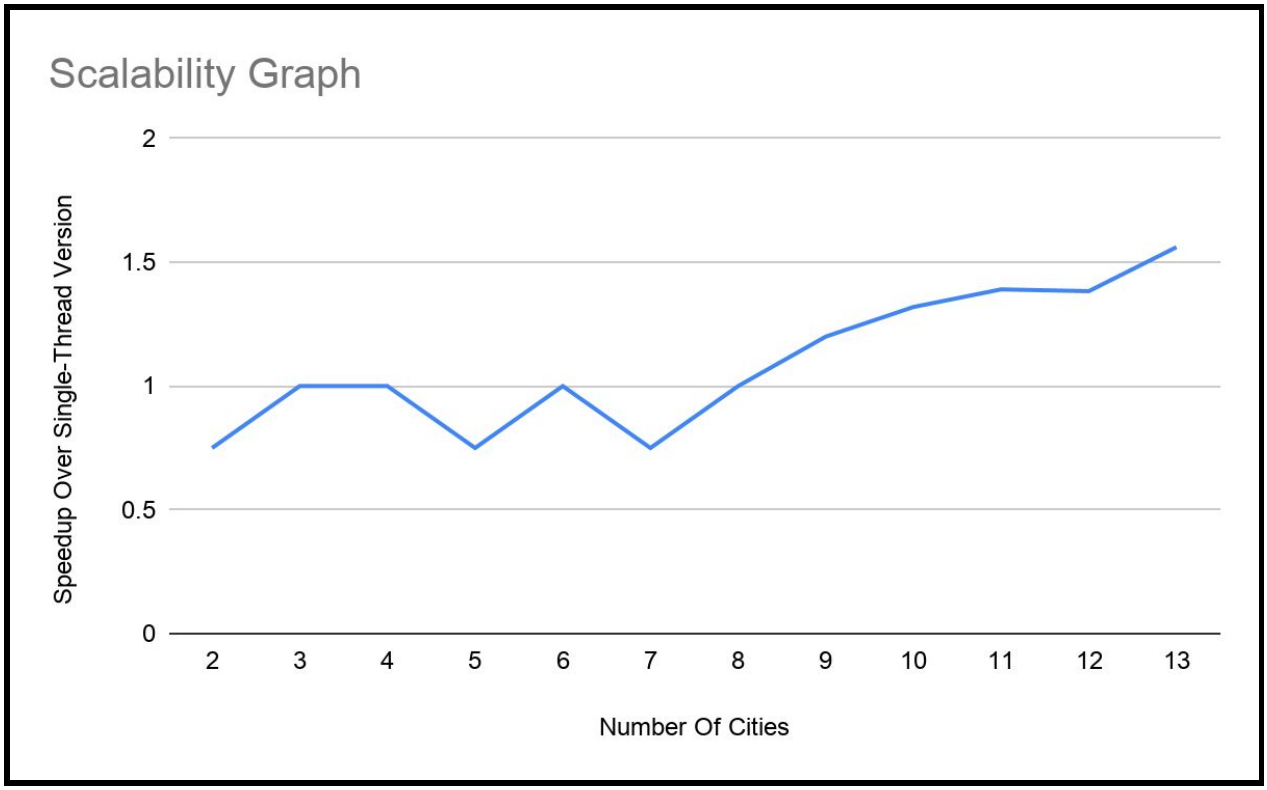
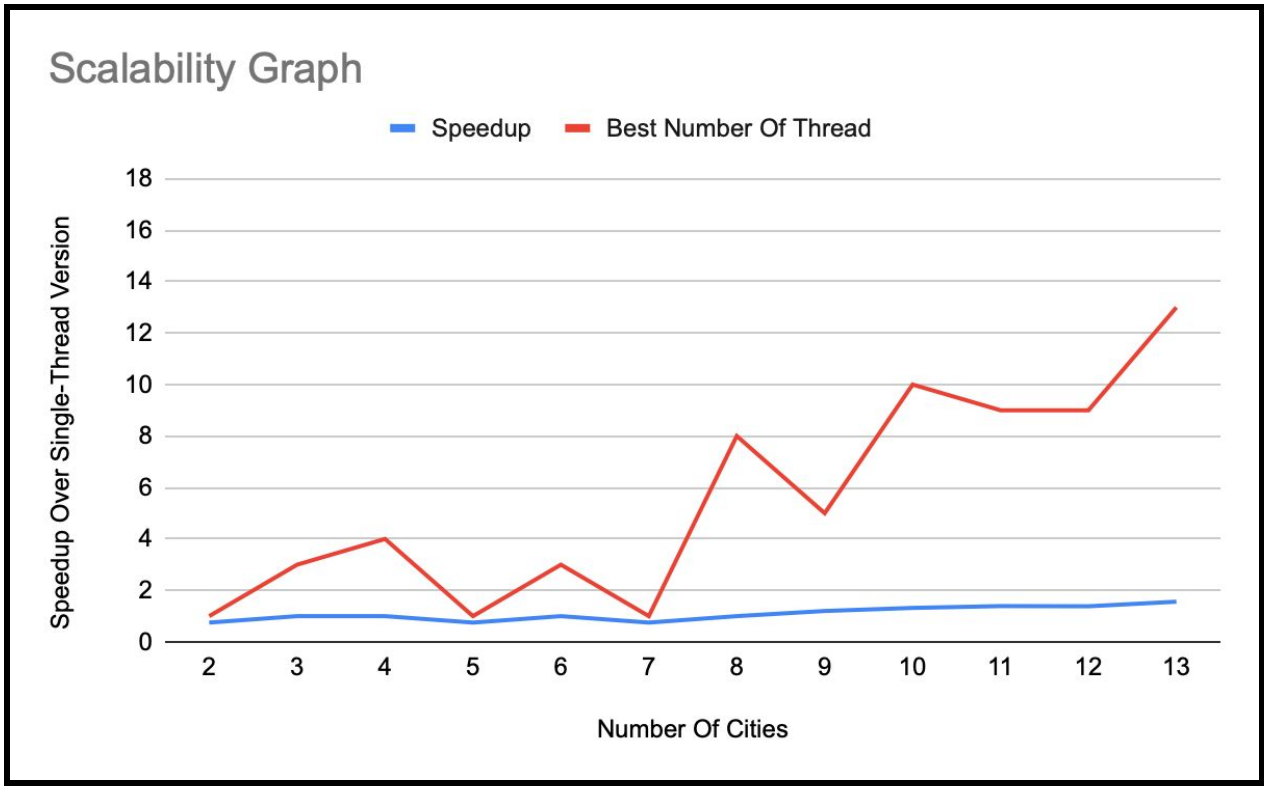
(wqg203) Berwin Gan
Parallel Computing Lab Assignment 2

Speedup Graph For Performance Of TSM



Graph Showing Relationship Between Speedup, Best Number Of Thread And Number Of Cities



(wqg203) Berwin Gan
Parallel Computing Lab Assignment 2

Table Of Speedup and Best Number Of Thread Per Solution

Clties	Speed Of 1 Thread	Best Speed Of X Thread That Is Not One	X Thread	Speedup	Best Number Of Thread
2	0.003	0.004	2	0.75	1
3	0.003	0.003	3	1	3
4	0.003	0.003	4	1	4
5	0.003	0.004	5	0.75	1
6	0.003	0.003	3	1	3
7	0.003	0.004	7	0.75	1
8	0.005	0.005	8	1	8
9	0.018	0.015	5	1.2	5
10	0.128	0.097	10	1.319587629	10
11	1.381	0.993	9	1.390735146	9
12	16.311	11.79	9	1.38346056	9
13	244.848	156.824	13	1.561291639	13

As I was using a lot of malloc, I could not pursue the problem further than 13 cities as I would run out of space come 14. As seen both in the graph and in the table above, as the problem set grew, the amount of speedup increase alongside it abit not in the same linear fashion. When the problem size was 8 and below, the overhead cost for creating multiple threads simply were to great that whatever sort of parallelization benefits obtained were small in comparison. Only after 9 cities did my program see any real speedup. I tried testing my program multiples times on different solutions with different number of threads but could not find any real pattern. The only thing that was clear was that for each solution, there was an optimal number of threads. It was not necessarily the case that the greater the number of threads the greater the speed up. It will approach a point where the creation of anymore threads would simple count more towards overhead than it will towards parallelization. Perhaps a conclusion that could be drawn from the graphs is that as the factorial problem size increases, the rate at which more threads will be needed to see an increase in speedup would far outpace the actual increase in speedup. However, as seen with this experiment, there would be a sweet spot. That said, from the table below, it is still not quite clear where the optimal number of threads would be for when N=12, it is near the top where T= 9 while for when N=13, T = 13.

(wqg203) Berwin Gan
Parallel Computing Lab Assignment 2

Solution to Cities12.txt			**Solution to Cities13.txt**		
Thread	Seconds	Speedup	Thread	Seconds	Speedup
1	16.311	1	1	244.848	1
2	13.861	1.176754924	2	247.743	0.9883145033
3	13.082	1.246827702	3	197.721	1.23835101
4	12.873	1.267070613	4	187.083	1.308766697
5	12.66	1.288388626	5	175.775	1.392962594
6	12.6	1.29452381	6	166.036	1.474668144
7	12.423	1.312967882	7	192.819	1.269833367
8	12.277	1.328581901	8	162.299	1.508622974
9	11.79	1.38346056	9	183.446	1.334714303
10	11.85	1.376455696	10	195.388	1.253137347
11	12.085	1.349689698	11	165.539	1.479095561
12	12.284	1.327824813	12	158.362	1.54612849
			13	156.824	1.561291639