Sanketh Karuturi
karutusa@oregonstate.edu

CS 575 Parallel Programming

Project #3
A Real Application Parallel Challenge

1. **Tell what machine you ran this on**

   → Flip Server

2. **Tell what operating system you were using**

   → Mac OS

3. **Tell what compiler you use**

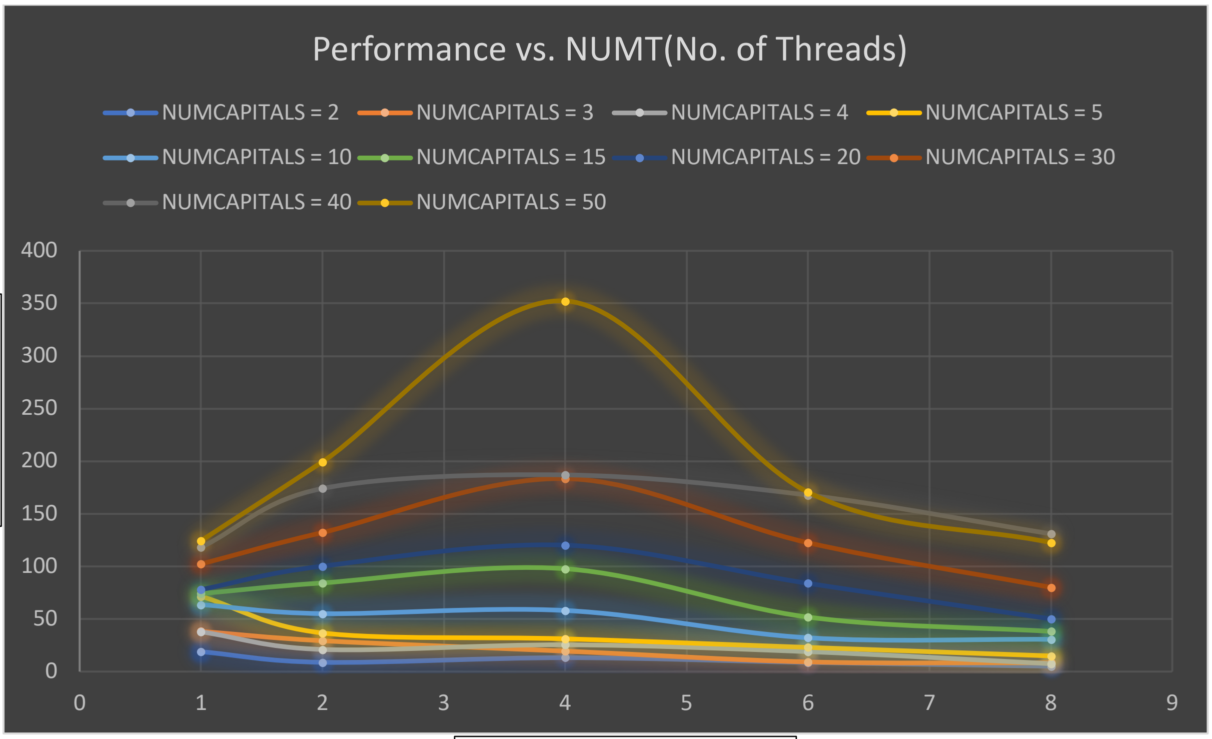
   → g++ compiler

4. **Include the table of performance data.**

NUMT	NUMCAPITALS	Performance
1	2	18.889
1	3	38.21
1	4	37.777
1	5	71.563
1	10	63.684
1	15	74.109
1	20	77.995
1	30	102.333
1	40	118.154
1	50	124.401
2	2	8.815
2	3	29.331
2	4	21.035
2	5	36.728
2	10	55.092
2	15	84.311
2	20	100.239
2	30	132.22
2	40	174.083
2	50	199.47

4	2	13.222
4	3	19.462
4	4	25.474
4	5	31.128
4	10	58.088
4	15	97.769
4	20	120.2
4	30	183.478
4	40	186.978
4	50	352.364
6	2	8.732
6	3	9.194
6	4	18.953
6	5	22.985
6	10	32.137
6	15	51.803
6	20	83.886
6	30	122.498
6	40	167.772
6	50	170.555
8	2	5.132
8	3	8.347
8	4	7.515
8	5	14.896
8	10	30.647
8	15	38.21
8	20	50.12
8	30	80.095
8	40	130.973
8	50	122.643

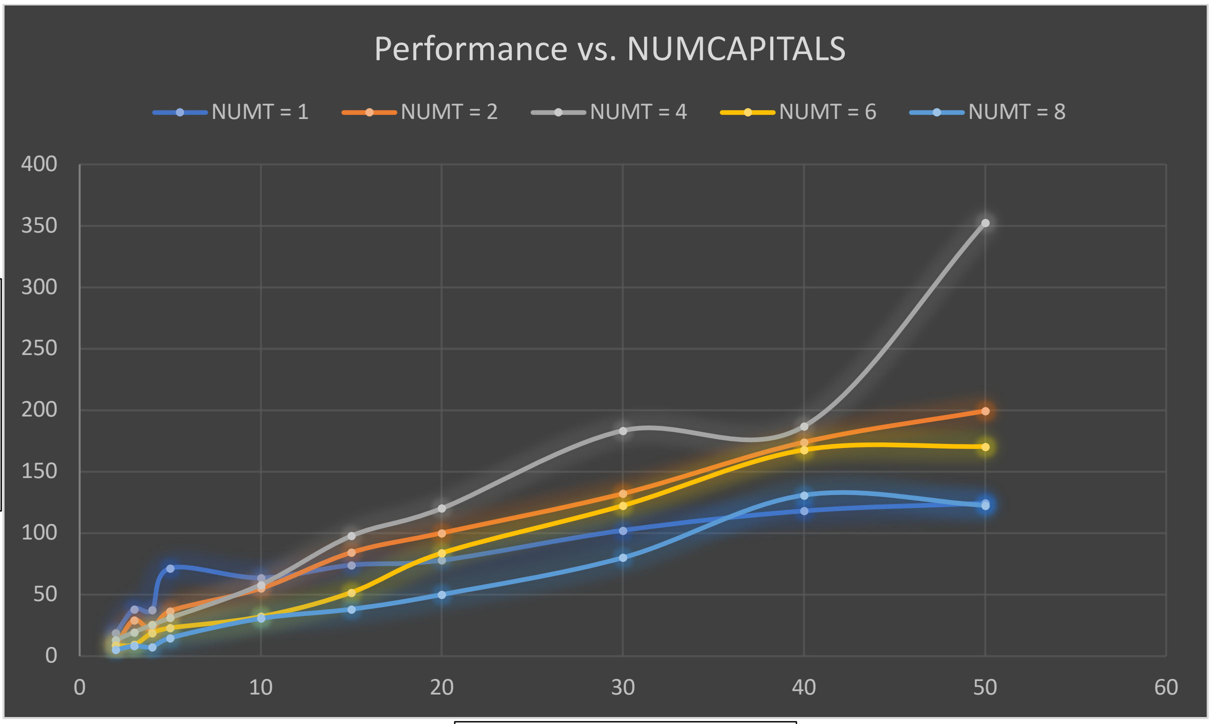
5. **Include a graph of performance vs. NUMT with the colored curves being NUMCAPITALS.**



PERFORMANCE

NUMBER OF THREADS

6. **Include a graph of performance vs. NUMCAPITALS cities with the colored curves being NUMT.**



PERFORMANCE

NUMBER OF CAPITALS

7. **Tell us what you discovered by doing this. What patterns are you seeing in the graphs?**

   I observed several key patterns and insights regarding the performance dynamics of parallel programming, especially as influenced by the number of threads (NUMT) and the number of capital cities (NUMCAPITALS):

Performance vs. NUMT (Number of Threads)

1. **Performance Improvement with More Threads**: The graph shows that performance generally improves (execution time decreases) as the number of threads increases, up to a certain point. This is expected in parallel computing, where distributing the workload across multiple threads can significantly reduce processing time.

2. **Diminishing Returns**: There is a clear point at which adding more threads yields diminishing returns. For example, the performance gain from increasing threads from 6 to 8 is less pronounced than from 1 to 4. This could be due to several factors such as overhead associated with managing more threads, or limitations due to the number of physical cores in the machine.

3. **Effect of NUMCAPITALS**: The impact of increasing NUMCAPITALS varies with the number of threads. For lower values of NUMCAPITALS, the increase in threads shows a consistent performance improvement. However, as NUMCAPITALS increases, the benefit of adding more threads diminishes more quickly, likely due to increased complexity and data management overhead.

Performance vs. NUMCAPITALS (Number of Capital Cities)

1. **Increasing Complexity**: As the number of capital cities increases, the execution time also increases across all thread configurations. This is expected since more capitals mean more distances to compute and more data points to cluster, increasing the computational load.

2. **Scaling with Threads**: The graph shows that higher numbers of threads manage the increase in NUMCAPITALS more efficiently than fewer threads. For instance, with 8 threads, the performance curve is flatter compared to just 1 or 2 threads, suggesting better handling of increased complexity due to parallel processing.

3. **Non-linear Growth**: The graph also indicates a non-linear increase in execution time as NUMCAPITALS grows, particularly noticeable in configurations with fewer threads. This non-linear growth likely reflects the non-linear increase in computational complexity as more data points (cities) must be processed repeatedly in the K-means algorithm.