Andrew Frost
CS415
PA4 Report
04/26/2017

Overview:

In this programming assignment matrices were multiplied using sequential and parallel algorithms. The sequential program made us of a simplistic matrix multiplication algorithm that ran in $O(n^3)$. Cannon's algorithm was used in the parallel program. In this report the parallel run times and sequential run times are compared. The complete data for this report can be found in data.xlsx. Table 1 shows a sample result of matrix multiplication.

Matrix Multiplication Results

Mat A:		
1	2	3
2	3	4
3	4	5
Mat B:		
3	2	1
4	3	2
5	4	3
Mat C:		
26	20	14
38	29	20
50	38	26

Table 1: an example of multiplying two 3x3 matrices using the sequential algorithm.

Table 1 shows a sample result from multiplying two matrices together using the sequential algorithm. The sequential algorithm will be discussed in depth below.

Sequential:

The sequential algorithm run times are based off of single sample run times. On select tests, multiple time trials were ran and little deviation was found. Based on the low variance found in select sample run times, the data used in this report for sequential run times was not averaged or processed. The sequential algorithm used had a time complexity of $O(n^3)$ and the run time results reflect this curve. Figure 1 graphs the run times of the sequential algorithm.

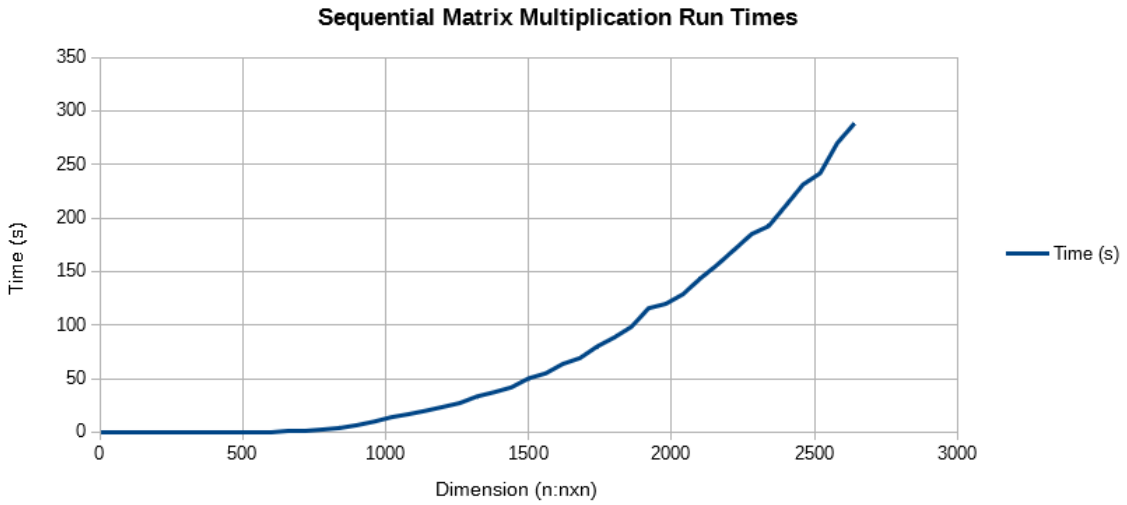

Figure 1: a graph of sequential run times in seconds. As the size of the matrix increases, the run time required increases dramatically.

As seen in the graph in Figure 1, the run time of the sequential algorithm increases dramatically as the matrix dimension size increases. The maximum tested dimension of 2640x2640 neared five minutes in run time. This not only shows the undesirability of an $O(n^3)$, but highlights the need for an efficient parallel algorithm. The run times of the sequential algorithm are included in Table 2.

Sequential Matrix Multiplication Run Times

Dimension (n:nxn)	Time (s)
5	1E-06
10	3E-06
12	4E-06
15	1E-05
20	2.1E-05
24	3.4E-05
25	7.2E-05
30	3.7E-05
35	9.9E-05
36	0.000108
40	8.1E-05
45	0.000208
48	0.000247
50	0.000208
55	0.000689
60	0.000475
120	0.002883
180	0.010299
240	0.018888
300	0.035875
360	0.066728
420	0.224465
480	0.204858
540	0.263546
600	0.360542
660	0.752537
720	0.730664
780	2.41055
840	3.80099
900	6.4185
960	9.80504
1020	13.8669
1080	16.7188
1140	19.9145
1200	23.4705
1260	27.1667
1320	33.3196
1380	37.2794
1440	41.7461
1500	50.1505
1560	54.8489
1620	63.6477
1680	69.1077
1740	79.9139
1800	88.3931
1860	98.3051
1920	115.69
1980	119.795
2040	128.712
2100	143.305
2160	156.224
2220	170.519
2280	184.983
2340	192.448
2400	211.715
2460	231.251
2520	241.846
2580	270.055
2640	288.443

Table 2: the run time in seconds of the sequential matrix multiplication algorithm. The lack of fluctuation can be attributed to consistency provided by the algorithm running only on a single processor core of the same make and model.

Nothing unusual was encountered when testing the sequential algorithm. As expected, the sequential algorithm followed an $O(n^3)$ curve as the size of the matrix increased. The sequential algorithm will now be compared to the parallel algorithm.

Parallel:

Changes made due to Critiques:

The following changes were made to the parallel code based on feedback from the critiques:

- The code responsible for shifting columns up and shifting rows left were moved into two functions: ShiftUp( ) and ShiftLeft( ).
- The extra shifts up and right done after the last multiplication have been removed through the use of an if statement.
- Modular has been added to the code by creating functions for specific tasks in main.
  - The file I/O in main has been moved into functions.
  - The printing of the answer has been moved into functions.
  - The sending and receiving of sub-matrices in the initialization has been moved into functions.
- Some comments in main have been modified to be more descriptive and meaningful.

The changes made from the critiques have greatly increased the readability, modularity, and efficiency of the code for the parallel implementation.

Run Times: