Group 12
Basics of Parallel Computing
Assignment 1

Members
Andrews Boateng Okyere
Edward Asante

Question 2.2: **Compute Speed-up and Parallel Effeciency for sizes(200,1000)**

	size	nprocs	mean_runtime	relative_speed_up	par.eff.
0	200	1	0.943135	1.000000	1.000000
1	200	2	0.512780	1.839260	0.919630
2	200	4	0.314576	2.998119	0.749530
3	200	8	0.218979	4.306958	0.538370
4	200	16	0.194960	4.837573	0.302348
5	200	24	0.172175	5.477786	0.228241
6	200	32	0.183329	5.144499	0.160766
7	1000	1	21.956377	0.042955	0.042955
8	1000	2	10.808621	0.087258	0.043629
9	1000	4	5.402483	0.174574	0.043644
10	1000	8	2.735586	0.344765	0.043096
11	1000	16	1.442968	0.653608	0.040850
12	1000	24	1.088874	0.866156	0.036090
13	1000	32	0.864803	1.090578	0.034081

Table 2.2

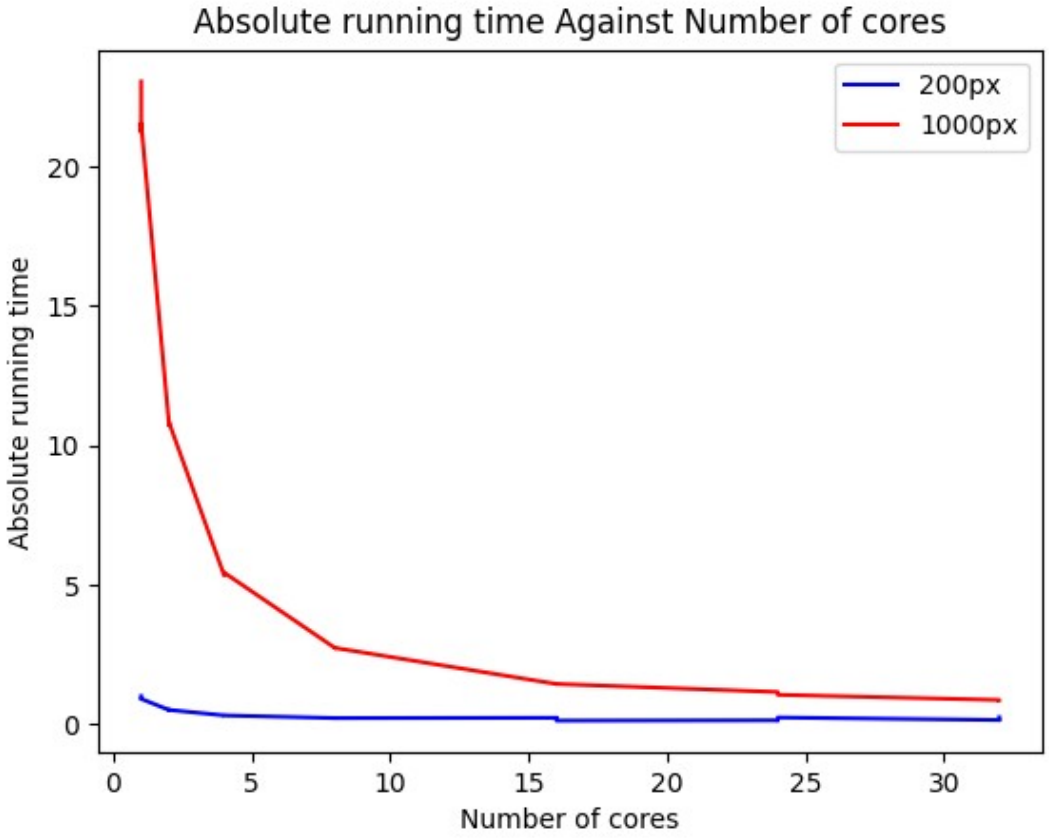

Fig 2.2.1

2.2.1

From figure 2.2.1, It can be seen that the absolute running time decreases as the number of cores increases. Also, the parallel code seems to only be effective when the number of cores is small.

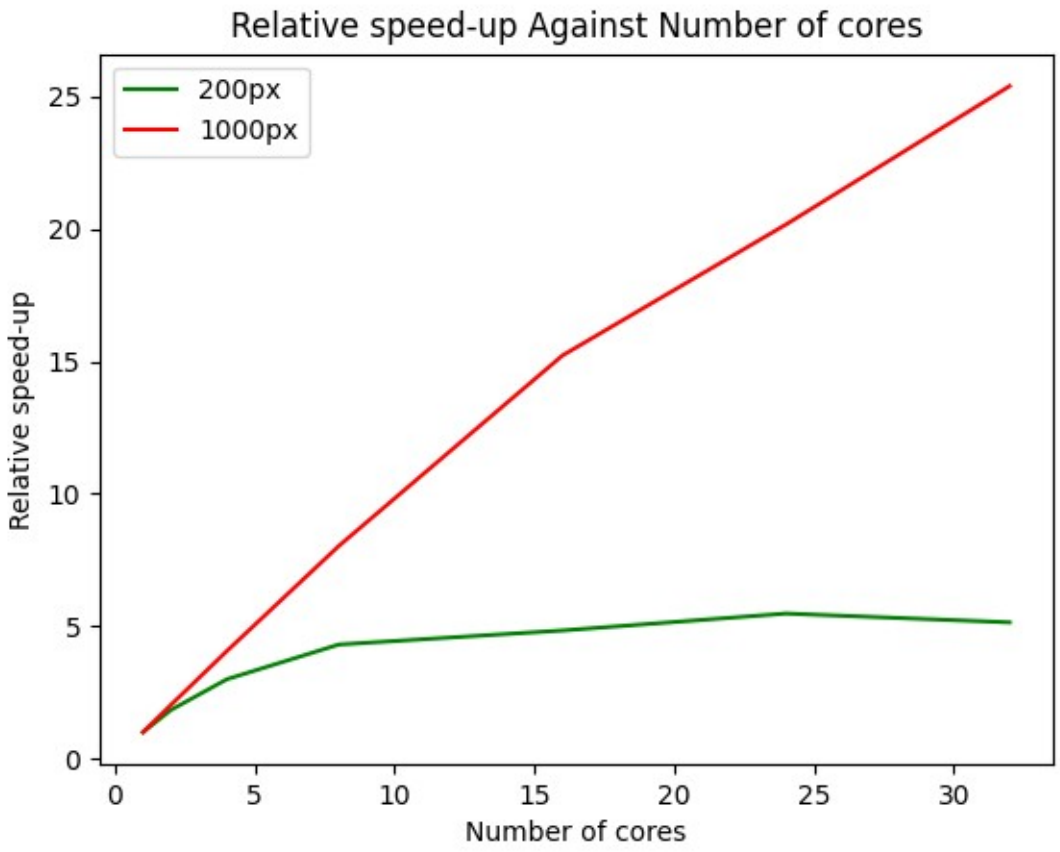

Fig 2.2.2

2.2.2

It can be observed from the graph that relative speed up generally increases as the number of cores increases. When the image size is smaller (200) the speed is lower compared to when the image size is bigger (1000).

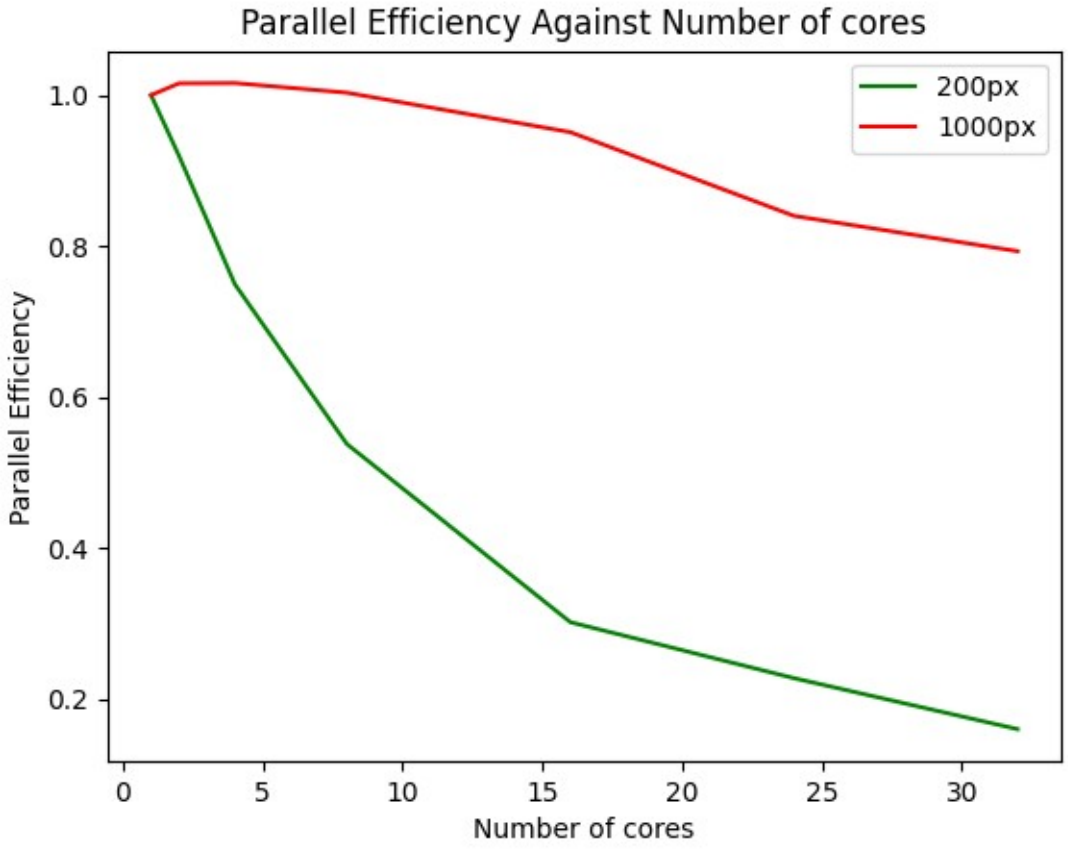

Fig 2.2.3

2.2.3

Due to the nature of the graph above, one can conclude that the parallel code used is not efficient. And that the efficiency decreases as the number of cores increases for both image sizes. Which assert that large number of cores doesn't necessary guarantee optimal work.

Question 2.3: Influence of Patch Size

	size	p	patch	mean_runtime(s)
0	1000	32	1	29.381266
1	1000	32	10	0.903133
2	1000	32	30	0.865266
3	1000	32	200	3.468138
4	1000	32	500	8.278478

Table 2.3

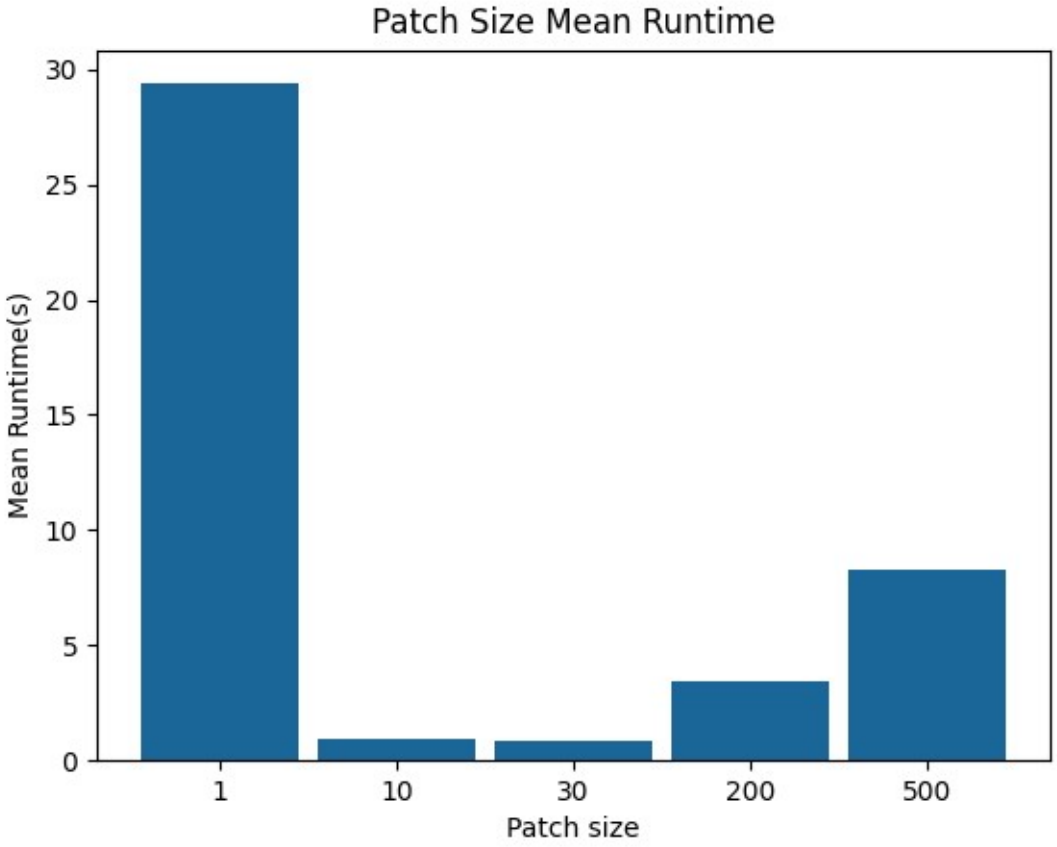

Fig 2.3

It is observed that when the patch size is 1 we obtained a highest Mean run time and least run time when the patch size is 30. The mean run time increases from patch size 10 to 500

Question 2.4: Finding the Best Patch Size

	size	p	patch	mean_runtime(s)
0	800	16	1	18.117337
1	800	16	2	4.668975
2	800	16	3	2.230116
3	800	16	4	1.542472
4	800	16	5	1.255630
5	800	16	6	1.152728
6	800	16	7	1.048376
7	800	16	8	1.081912
8	800	16	9	1.076364
9	800	16	10	1.009167
10	800	16	11	0.939475
11	800	16	12	0.972861
12	800	16	13	0.973875
13	800	16	14	0.973715
14	800	16	15	0.971881
15	800	16	16	1.005263
16	800	16	17	0.940152
17	800	16	18	0.971858
18	800	16	19	0.938615
19	800	16	20	1.004800
20	800	16	21	0.937735
21	800	16	22	1.038185
22	800	16	23	1.038178
23	800	16	24	0.971328
24	800	16	25	0.971660
25	800	16	26	1.037941
26	800	16	27	1.004797
27	800	16	28	1.037341
28	800	16	29	1.037807
29	800	16	30	1.037936

Table 2.4

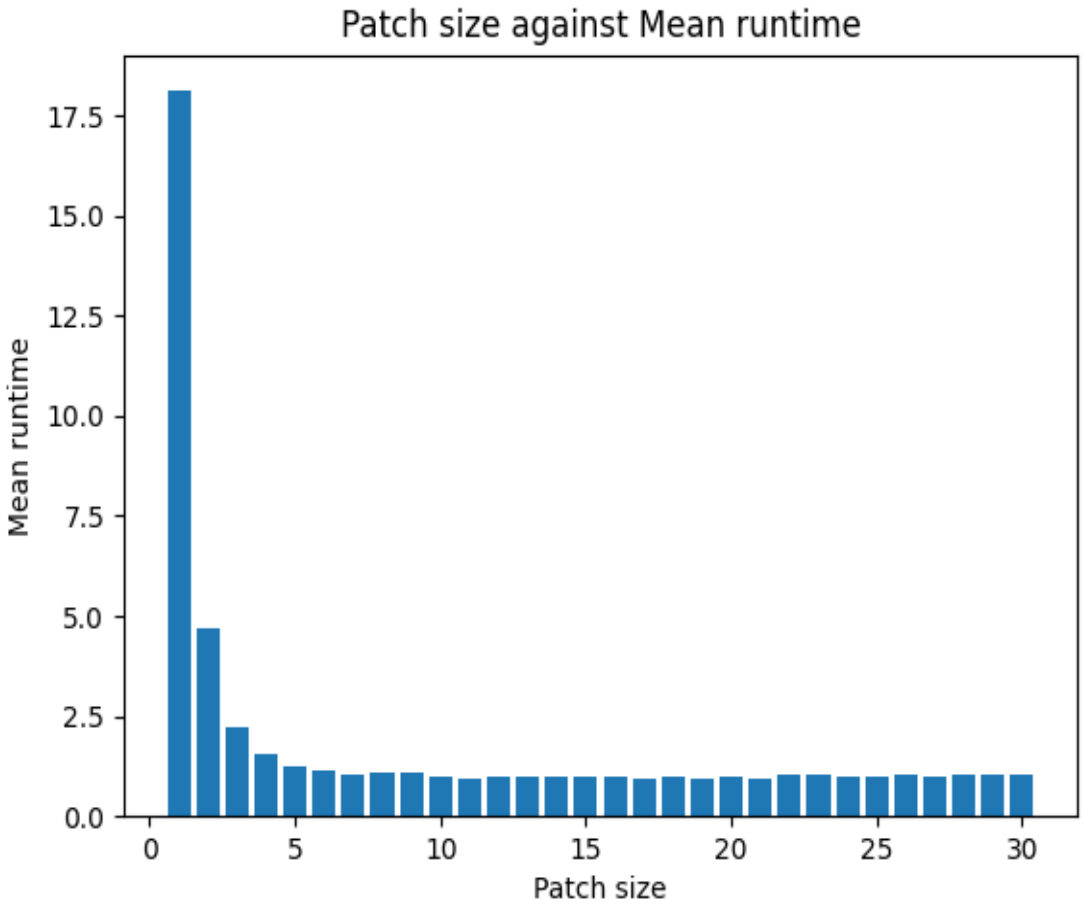

Fig 2.4

Observation:
Clearly, it can be observed that patch size is inversely proportional to the average runtime of the code. And that average mean time decreases as the number of patch increases.