4F03 Report

Jean Lucas Ferreira (1152120)
George Plukov (1316246)

April 2017

1 Analysis Report

The following data was gathered with four different image sizes when a blur radius of 15 pixels was applied. When timing the program, we created a barrier after the basic MPI initialization, then proceeded to start an elapsed time for each process. After all processes were finished we used MPI_Reduce() to find the process that had the longest elapsed time.

Time listed in seconds

Processes	Image Size (blur radius= 15)			
	960x720 (snail)	1920x1080 (fox)	3000x2000 (wolf)	4400x2800 (bird)
1	5.004	19.612	42.537	143.210
2	2.640	12.639	22.138	80.251
8	0.884	2.371	6.615	14.907
16	1.068	2.057	4.450	8.125
32	1.555	1.623	4.159	6.970

2 Speedup

Listed in the below chart is the speedup experienced by running the program on the respective amount of processes.

Processes	Image Size (blur radius= 15)			
	960x720 (snail)	1920x1080 (fox)	3000x2000 (wolf)	4400x2800 (bird)
1	-	-	-	-
2	1.854	1.552	1.92	2.244
4	3.4	4.66	2.76	4.189
8	5.66	8.27	6.43	7.636
16	4.68	9.534	9.56	11.454
32	3.215	12.084	10.228	15.655

3 Efficiency

Processes	Image Size (blur radius= 15)			
	960x720 (snail)	1920x1080 (fox)	3000x2000 (wolf)	4400x2800 (bird)
1	1	1	1	1
2	0.948	0.776	0.961	0.890
4	0.849	1.116	0.69	1.512
8	0.708	1.034	0.804	1.200
16	0.293	0.596	0.597	1.102
32	0.100	0.378	0.320	0.649

4 Summary

As we can see from the respective tables, for smaller images, increasing the number of process can reduce performance. This is due to the fact that we create more processes than needed, and results in unnecessary communication overhead.

In addition, as we can see from the Efficiency table, we get values that are theoretically impossible. The result of this has been found to be issues in a real-world scenario. Since our program deploys processes on different servers across the school's network, it is possible that some of the execution time is delayed due to heavy traffic.

Another factor in calculating run times is which processors the program is distributed to. Depending on the clock speed of the selected processors significantly different run times can be experienced.

We have ran many test cases with the large image files (Fox, Wolf and Bird) and have observed that timing results for a serial program (one process) can fluctuate by up to 40%. One way to reduce the error the server causes would be to run a large amount of each test and average the values, once this is done all efficiency values would ideally be less than or equal to one.

In regards to scalability, our program is *weakly scalable*. Since we can maintain a constant efficiency with an increasing problem size and increasing number of processes. It is *not* strongly scalable because

This can be seen

5 Results

Figure 1: Snail 960x720

Figure 2: Fox 1920x1080

Figure 3: Wolf 3000x2000

Figure 4: Bird 4400x2800